Demand Forecasting in a Supply Chain

A Machine Learning Based Data Analytic Approach Using R

Ujjal Kumar Mukherjee
Assistant Professor
Business Administration
University of Illinois, Urbana-Champaign

March 30, 2016

1 Introduction to Forecasting

Managing supply chains is primarily about managing uncertainties in supply chains. One of the primary source of uncertainty is demand. Demand uncertainties lead to inefficiencies in the supply chain such as excess supply or lack of supply. Excess supply of products in the supply chain can lead to losses due to the sunk cost associated with unsold inventory. Similarly, lack of adequate supply of products leads to losses due to the opportunity cost of lost sales. There may be other sorts of uncertainties associated with a supply chain such as supply side uncertainties, price uncertainty, currency value uncertainty for international operations and supply chains, and many more. Hence, it is critical to be able to forecast the future. A forecast is a statement about the likely future value of a variable of interest, such as demand.

Forecasts help in reducing uncertainties from a supply chain. A good forecast helps in planning activities in a supply chain such as capacity investments, production plans, delivery plans and schedules, ordering cycles and inventory planning. Consider the forecasting problem for a company like Amazon. Amazon sells thousands of items on-line from its warehouses. Amazon needs to stock many of these items in its warehouses for timely delivery of orders. Also, Amazon has several regional warehouses to optimize delivery costs and time. To be effective, Amazon would need to forecast the demand for these items that it delivers through its own warehouses. Amazon would need to forecast at various levels for managing its operations. Firstly, Amazon would need to generate a national forecast, which it would need to breakdown into regional forecasts for stocking of the items in the right amount. Also, Amazon would need to forecast the mix of products at the regional level. As an illustration, Amazon would need to forecast how many televisions of which brand (LG, Samsung, panasonic, etc.) of what type (LCD, LED, 3D, flat, curved, etc.) of what specifications (dimensions) it is likely to sell in each region in the coming year. Also, Amazon would need to estimate the seasonality of demand to manage its orders and stocking. Forecasting can be done using past data related to the variable of interest and projecting the past data into the future, or forecasting can be done by associating the variable of interest such as demand to other variables such as GDP growth, job growth, lending rates, currency values, and other macroeconomic indicators and using the association relationship to estimate the likely demand distribution in the future.

2 Forecasting Methods

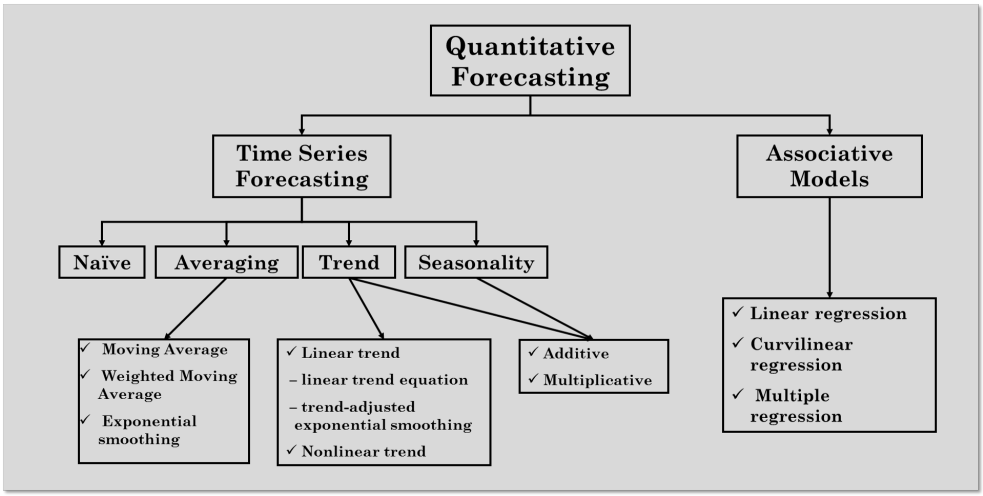
Demand forecasting methods can be broadly divided into two classes of methods, namely, (i) **Time Series Based Forecasting**, and (ii) **Causal Forecasting using Associative Models**. In standard time series based forecasting methods, purely past demand data is used to forecast future demand using time series

methods. In causal forecasting, past demand data along with data related to factors, such as economic factors, environmental factors, market related factors or consumer behavior related factors that influence market demand for a product or a service are used to build functional relationships between demand and the explanatory factors. The focus of this tutorial is primarily causal forecasting. However, first I provide an overview of standard time series based forecasting methods using excel spreadsheet, and then we move on to causal forecasting using regression models and machine learning (data mining) models in R.

3 Standard Time Series Based Forecasting Methods (Using Excel Spreadsheets)

As mentioned earlier, demand forecasting methods can be divided into two broad categories, i.e., *time series forecasting* and *associative models*. Again both these methods of forecasting can be of several types. Time series forecasting depends on the past data related to the variable of interest such as demand. The implicit assumption in time series forecasting is that the past is representative of the future. Hence, time series based forecasting works best for stable mature markets. However, many products markets may be new and the past may not be a very good representative of the future. In such cases, using causal models or associative models such as linear and non-linear regression models may be more appropriate. There is no single method which works best for all situations. It is important to select the most appropriate method of forecasting depending on the situation and variable being forecast. Figure 1 provides an overview of the various standard methods of forecasting available.

Figure 1: **Standard Forecasting Methods**



3.1 Time Series Forecasting with Seasonality

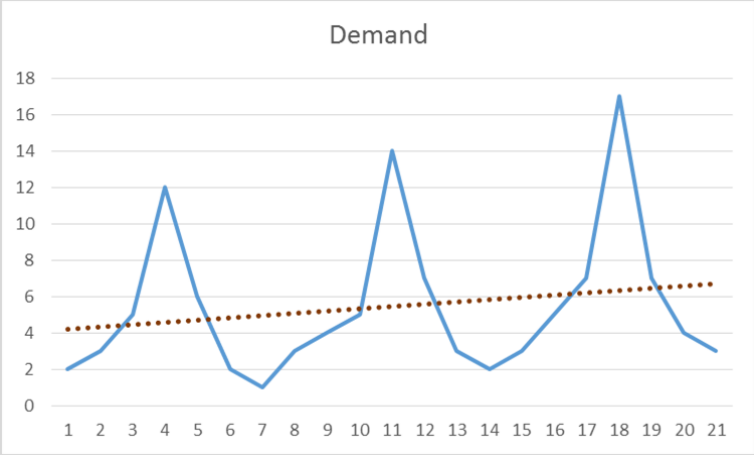
Consider the following data shown in figure 2. (Note: All examples shown here are done in MS excel).

Figure 2: **Daily Demand Data for a Product**

Week	Day	Period (Days)	Demand
1	Sun	1	2
1	Mon	2	3
1	Tue	3	5
1	Wed	4	12
1	Thur	5	6
1	Fri	6	2
1	Sat	7	1
2	Sun	8	3
2	Mon	9	4
2	Tue	10	5
2	Wed	11	14
2	Thur	12	7
2	Fri	13	3
2	Sat	14	2
3	Sun	15	3
3	Mon	16	5
3	Tue	17	7
3	Wed	18	17
3	Thur	19	7
3	Fri	20	4
3	Sat	21	3

At the beginning, it is a good idea to visualize the data in a graph as shown in figure 3.

Figure 3: **Daily Demand Data Graph for a Product**



From the graph we can observe the following.

1. There is seasonality in the demand data. The demand for the product is lower during the weekends than the demand during weekdays. Also, during the weekdays, the demand peaks at the middle of the week.
2. The pattern of demand is weekly and is repeating every 7 days. Hence, the *periodicity* is 7 days.
3. There seems to be an overall increasing trend in the demand from week to week.

The first method that we will use to forecast the demand for the next week is as follows:

1. Deseasonalize the demand by averaging and estimate the trend using a linear regression.
2. Estimate seasonal factors.
3. Use the trend equation from the linear regression to forecast the deseasonalized demand for the next 7 days.
4. Multiply the deseasonalized demand with the seasonal factors to get the final forecast for the next week.

Step 1. Deseasonalize the demand: Deseasonalized demand is the estimated demand that we would have observed had there been no seasonality in the demand pattern. To deseasonalize the demand for a specific period we would average the demand over the periodicity of the demand with the specific period being in the center of the period. For example, if we are estimating the deseasonalized demand for the period 10, since the periodicity of the demand is 7, we would average the demand from period 7 to period 13. Observe that the period 10 lies in the center of the periods that are being used for the averaging. Hence, we would start from period 4 and estimate the deseasonalized demand up to period 18. So the deseasonalized demand for period 4 would be $\bar{d}_4 = (d_1+d_2+d_3+d_4+d_5+d_6+d_7)/7 = (2+3+5+12+6+2+1)/7 = 4.4285$. The deseasonalized demand for the demand data is shown below in figure 4.

Figure 4: **Deseasonalizing Demand Data**

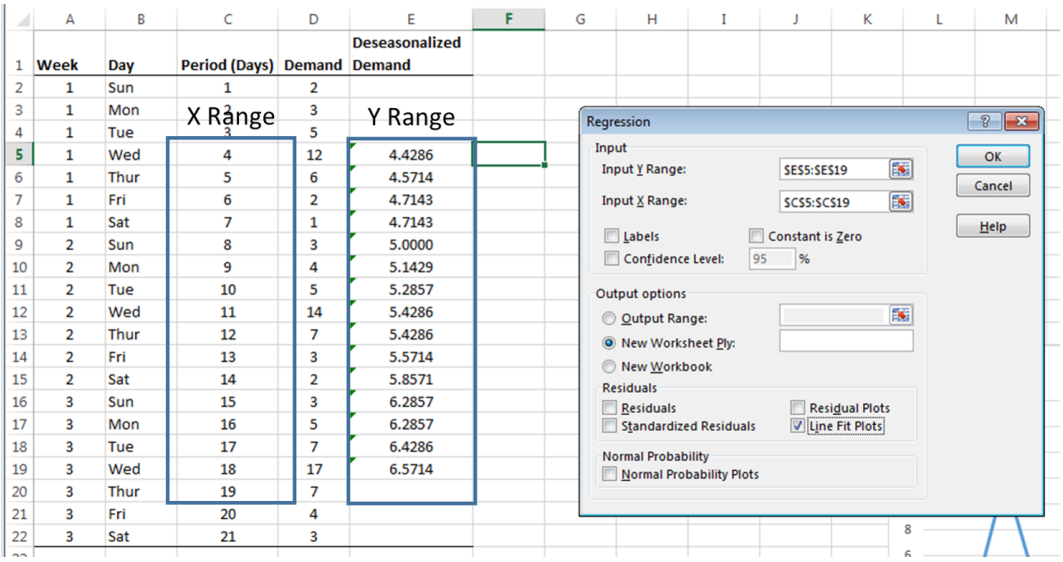
Week	Day	Period (Days)	Demand	Deseasonalized Demand
1	Sun	1	2	
1	Mon	2	3	
1	Tue	3	5	
1	Wed	4	12	4.4286
1	Thur	5	6	4.5714
1	Fri	6	2	4.7143
1	Sat	7	1	4.7143
2	Sun	8	3	5.0000
2	Mon	9	4	5.1429
2	Tue	10	5	5.2857
2	Wed	11	14	5.4286
2	Thur	12	7	5.4286
2	Fri	13	3	5.5714
2	Sat	14	2	5.8571
3	Sun	15	3	6.2857
3	Mon	16	5	6.2857
3	Tue	17	7	6.4286
3	Wed	18	17	6.5714
3	Thur	19	7	
3	Fri	20	4	
3	Sat	21	3	

Observe that the periodicity of seasonality is 7, which is an odd number. Since,

the periodicity is an odd number it is possible to get an exact middle period. However, if the periodicity is even, then it is not possible to get an exact middle period. For example is the periodicity is 4, then the center lies between 3 and 4. We would need to do the averaging slightly differently for deseasonalizing the demand when the periodicity is even. This is explained in details in a subsequent example.

Step 2. Trend Line Estimation from the Deseasonalized Data: After deseasonalizing the demand data we would need to estimate the trend of the demand using a linear regression fit. The trend can be non-linear, however, for small data it is often not feasible to fit non-linear trends. The trend is estimated in excel in the method shown in figure 5.

Figure 5: **Regression Estimation in Excel**



	A	B	C	D	E
1	Week	Day	Period (Days)	Demand	Deseasonalized Demand
2	1	Sun	1	2	
3	1	Mon	2	3	
4	1	Tue	3	5	
5	1	Wed	4	12	4.4286
6	1	Thur	5	6	4.5714
7	1	Fri	6	2	4.7143
8	1	Sat	7	1	4.7143
9	2	Sun	8	3	5.0000
10	2	Mon	9	4	5.1429
11	2	Tue	10	5	5.2857
12	2	Wed	11	14	5.4286
13	2	Thur	12	7	5.4286
14	2	Fri	13	3	5.5714
15	2	Sat	14	2	5.8571
16	3	Sun	15	3	6.2857
17	3	Mon	16	5	6.2857
18	3	Tue	17	7	6.4286
19	3	Wed	18	17	6.5714
20	3	Thur	19	7	
21	3	Fri	20	4	
22	3	Sat	21	3	

The regression estimation results are shown in figure 6.

Figure 6: **Regression Estimation Results in Excel**

SUMMARY OUTPUT								
Regression Statistics								
Multiple R	0.98957459							
R Square	0.979257869							
Adjusted R Square	0.97766232							
Standard Error	0.105795647							
Observations	15							
ANOVA								
	df	*SS*	*MS*	*F*	*Significance F*			
Regression	1	6.869460641	6.869460641	613.7436968	2.51253E-12			
Residual	13	0.145505345	0.011192719					
Total	14	7.014965986						
	Coefficients	*Standard Error*	*t Stat*	*P-value*	*Lower 95%*	*Upper 95%*	*Lower 95.0%*	*Upper 95.0%*
Intercept	**3.724659864**	0.074719708	49.84842641	3.11917E-16	3.563237749	3.886081979	3.563237749	3.886081979
X Variable 1	**0.156632653**	0.006322499	24.77385107	2.51253E-12	0.142973724	0.170291582	0.142973724	0.170291582

The coefficient estimation for *Intercept* and *X Variable 1* are of interest to us. This result indicates that the deseasonalized demand is a linear function of the time period (lets say T). The deseasonalized demand equation is $\bar{d} = 3.7246 + 0.1566 \times T$. We would use this regression equation to predict the deseasonalized demand for all the periods in the data including the additional 7 days period for which we would need to do the forecast. So for period 1 the forecast deseasonalized demand is $\hat{d}_1 = 3.7246 + 0.1566 \times 1 = 3.8813$ and the forecast deseasonalized demand for period $T = 10$ will be $\hat{d_{10}} = 3.7246 + 0.1566 \times 10 = 5,2910$ and so on. These are called the forecast deseasonalized demand or the fitted deseasonalized demand. The fitted deseasonalized demand is shown in figure 7.

Figure 7: **Fitted Deseasonalized Demand from Regression Estimates**

Week	Day	Period (Days)	Demand	Deseasonalized Demand	Fitted Deseasonalzied Demand
1	Sun	1	2		3.8813
1	Mon	2	3		4.0379
1	Tue	3	5		4.1946
1	Wed	4	12	4.4286	4.3512
1	Thur	5	6	4.5714	4.5078
1	Fri	6	2	4.7143	4.6645
1	Sat	7	1	4.7143	4.8211
2	Sun	8	3	5.0000	4.9777
2	Mon	9	4	5.1429	5.1344
2	Tue	10	5	5.2857	5.2910
2	Wed	11	14	5.4286	5.4476
2	Thur	12	7	5.4286	5.6043
2	Fri	13	3	5.5714	5.7609
2	Sat	14	2	5.8571	5.9175
3	Sun	15	3	6.2857	6.0741
3	Mon	16	5	6.2857	6.2308
3	Tue	17	7	6.4286	6.3874
3	Wed	18	17	6.5714	6.5440
3	Thur	19	7		6.7007
3	Fri	20	4		6.8573
3	Sat	21	3		7.0139
4	**Sun**	**22**			**7.1706**
4	**Mon**	**23**			**7.3272**
4	**Tue**	**24**			**7.4838**
4	**Wed**	**25**			**7.6405**
4	**Thur**	**26**			**7.7971**
4	**Fri**	**27**			**7.9537**
4	**Sat**	**28**			**8.1104**

Step 3. Estimating the Seasonality Factors Associated with the Weekdays: The fitted deseasonalized demand helps in estimating the *seasonality factors* associated with each day of the week in this case. The seasonality factor represents the impact of the specific period on the demand and can be thought of as a multiplicative factor on the deseasonalized forecast. The interpretation of the seasonality factor is how much the demand is shrunk or inflated due to the effect of seasonality in a specific period. The seasonality factor is estimated as a ratio of the actual demand and the fitted deseasonalized demand. So seasonality factor associated with period i is given by equation 1.

$$S_i = \frac{\text{Actual observed demand}}{\text{Fitted Deseasonalized Demand}} = \frac{d_i}{\hat{d}_i} \tag{1}$$

The estimation of the seasonality factors is shown in 8.

Figure 8: **Computation of Seasonality Factors**

Week	Day	Period (Days)	Demand	Deseasonalized Demand	Fitted Deseasonalzied Demand	Seasonality Factor
1	Sun	1	2		3.8813	0.5153
1	Mon	2	3		4.0379	0.7430
1	Tue	3	5		4.1946	1.1920
1	Wed	4	12	4.4286	4.3512	2.7579
1	Thur	5	6	4.5714	4.5078	1.3310
1	Fri	6	2	4.7143	4.6645	0.4288
1	Sat	7	1	4.7143	4.8211	0.2074
2	Sun	8	3	5.0000	4.9777	0.6027
2	Mon	9	4	5.1429	5.1344	0.7791
2	Tue	10	5	5.2857	5.2910	0.9450
2	Wed	11	14	5.4286	5.4476	2.5699
2	Thur	12	7	5.4286	5.6043	1.2491
2	Fri	13	3	5.5714	5.7609	0.5208
2	Sat	14	2	5.8571	5.9175	0.3380
3	Sun	15	3	6.2857	6.0741	0.4939
3	Mon	16	5	6.2857	6.2308	0.8025
3	Tue	17	7	6.4286	6.3874	1.0959
3	Wed	18	17	6.5714	6.5440	2.5978
3	Thur	19	7		6.7007	1.0447
3	Fri	20	4		6.8573	0.5833
3	Sat	21	3		7.0139	0.4277
4	**Sun**	**22**			**7.1706**	
4	**Mon**	**23**			**7.3272**	
4	**Tue**	**24**			**7.4838**	
4	**Wed**	**25**			**7.6405**	
4	**Thur**	**26**			**7.7971**	
4	**Fri**	**27**			**7.9537**	
4	**Sat**	**28**			**8.1104**	

Notice that the seasonality factors associated with each day of the week for different weeks are similar but not exactly the same. To get one seasonality factor for each day of the week we would take an average of the seasonality factors for a day from all three weeks. For example, the seasonality factor associated with Sunday will be $S_{Sun} = (0.5153 + 0.6027 + 0.4939)/3 = 0.5373$. The seasonality factors associated with each day is given in figure 9.

Figure 9: **Computation of Seasonality Factors**

Day	Seasonality Factor
Sun	0.5373
Mon	0.7748
Tue	1.0776
Wed	2.6419
Thur	1.2082
Fri	0.5109
Sat	0.3244

Step 4. Computing the Final Forecast: The final step of forecasting is to estimate the demand using the fitted deseasonalized demand ($\hat{d}_i$) and the estimated seasonality factor (S_i). The final demand $\hat{D}_i$ or period i is given by equation 2.

$$\hat{D}_i = \hat{d}_i \times S_i \tag{2}$$

Figure 10 gives the final forecast for the observed periods and the additional 7 days beyond the observed periods.

Figure 10: **Final Demand Forecast**

Week	Day	Period (Days)	Demand	Deseasonalized Demand	Deseasonalzied Demand	Seasonality Factor	Forecast Demand
1	Sun	1	2		3.8813	0.5153	2.0854
1	Mon	2	3		4.0379	0.7430	3.1287
1	Tue	3	5		4.1946	1.1920	4.5202
1	Wed	4	12	4.4286	4.3512	2.7579	11.4952
1	Thur	5	6	4.5714	4.5078	1.3310	5.4466
1	Fri	6	2	4.7143	4.6645	0.4288	2.3833
1	Sat	7	1	4.7143	4.8211	0.2074	1.5638
2	Sun	8	3	5.0000	4.9777	0.6027	2.6745
2	Mon	9	4	5.1429	5.1344	0.7791	3.9782
2	Tue	10	5	5.2857	5.2910	0.9450	5.7018
2	Wed	11	14	5.4286	5.4476	2.5699	14.3918
2	Thur	12	7	5.4286	5.6043	1.2491	6.7713
2	Fri	13	3	5.5714	5.7609	0.5208	2.9435
2	Sat	14	2	5.8571	5.9175	0.3380	1.9195
3	Sun	15	3	6.2857	6.0741	0.4939	3.2636
3	Mon	16	5	6.2857	6.2308	0.8025	4.8278
3	Tue	17	7	6.4286	6.3874	1.0959	6.8834
3	Wed	18	17	6.5714	6.5440	2.5978	17.2885
3	Thur	19	7		6.7007	1.0447	8.0961
3	Fri	20	4		6.8573	0.5833	3.5037
3	Sat	21	3		7.0139	0.4277	2.2751
4	**Sun**	**22**			**7.1706**		**3.8527**
4	**Mon**	**23**			**7.3272**		**5.6773**
4	**Tue**	**24**			**7.4838**		**8.0649**
4	**Wed**	**25**			**7.6405**		**20.1851**
4	**Thur**	**26**			**7.7971**		**9.4208**
4	**Fri**	**27**			**7.9537**		**4.0640**
4	**Sat**	**28**			**8.1104**		**2.6308**

3.2 Understanding Forecast Accuracy

The best way to test forecast accuracy is to forecast for some periods ahead and wait to observe the actual demand for the forecast period and measure the accuracy using a deviation measure. However, in the absence of actual observations for the forecast period, the accuracy of forecast can be computed using the observation that have been used for forecasting. Commonly there are two types of forecast accuracy measures, namely, Mean Absolute Deviation (MAD) and Mean Square Error (MSE). If there are n observed periods and the observed demand for period i is D_i and the forecast demand for period i is $\hat{D}_i$, then the mean absolute deviation (MAD) measure is given by equation 3 and mean square error (MSE) is given by equation 4. The absolute value and the squaring operation is done to remove the mutual error canceling effect in the presence of both positive errors and negative errors.

$$\mathbf{MAD} = \frac{1}{n}\sum_{i=1}^{n}|D_i - \hat{D}_i| \tag{3}$$

$$\mathbf{MSE} = \frac{1}{n}\sum_{i=1}^{n}(D_i - \hat{D}_i)^2 \tag{4}$$

The errors for the forecast above 10 is shown in table 11.

Figure 11: **Final Demand Forecast and Forecast Errors**

Week	Day	Period (Days)	Demand	Deseasonalized Demand	Deseasonalzied Demand	Seasonality Factor	Forecast Demand	MAD	MSE
1	Sun	1	2		3.8813	0.5153	2.0854	0.0854	0.0073
1	Mon	2	3		4.0379	0.7430	3.1287	0.1287	0.0166
1	Tue	3	5		4.1946	1.1920	4.5202	0.4798	0.2302
1	Wed	4	12	4.4286	4.3512	2.7579	11.4952	0.5048	0.2548
1	Thur	5	6	4.5714	4.5078	1.3310	5.4466	0.5534	0.3063
1	Fri	6	2	4.7143	4.6645	0.4288	2.3833	0.3833	0.1469
1	Sat	7	1	4.7143	4.8211	0.2074	1.5638	0.5638	0.3179
2	Sun	8	3	5.0000	4.9777	0.6027	2.6745	0.3255	0.1060
2	Mon	9	4	5.1429	5.1344	0.7791	3.9782	0.0218	0.0005
2	Tue	10	5	5.2857	5.2910	0.9450	5.7018	0.7018	0.4925
2	Wed	11	14	5.4286	5.4476	2.5699	14.3918	0.3918	0.1535
2	Thur	12	7	5.4286	5.6043	1.2491	6.7713	0.2287	0.0523
2	Fri	13	3	5.5714	5.7609	0.5208	2.9435	0.0565	0.0032
2	Sat	14	2	5.8571	5.9175	0.3380	1.9195	0.0805	0.0065
3	Sun	15	3	6.2857	6.0741	0.4939	3.2636	0.2636	0.0695
3	Mon	16	5	6.2857	6.2308	0.8025	4.8278	0.1722	0.0297
3	Tue	17	7	6.4286	6.3874	1.0959	6.8834	0.1166	0.0136
3	Wed	18	17	6.5714	6.5440	2.5978	17.2885	0.2885	0.0832
3	Thur	19	7		6.7007	1.0447	8.0961	1.0961	1.2014
3	Fri	20	4		6.8573	0.5833	3.5037	0.4963	0.2463
3	Sat	21	3		7.0139	0.4277	2.2751	0.7249	0.5254
4	**Sun**	**22**			**7.1706**		**3.8527**		
4	**Mon**	**23**			**7.3272**		**5.6773**		
4	**Tue**	**24**			**7.4838**		**8.0649**		
4	**Wed**	**25**			**7.6405**		**20.1851**		
4	**Thur**	**26**			**7.7971**		**9.4208**		
4	**Fri**	**27**			**7.9537**		**4.0640**		
4	**Sat**	**28**			**8.1104**		**2.6308**		
							Mean	**0.3649**	**0.2030**

So the MAD for the forecast method we have used above is 0.3649 and the MSE is 0.2030. The error measures MAD or MSE are used for comparing between different forecasting methods. Lower the forecast error the better it is. So if there are several different forecast methods available, we should choose the one that has the minimum MAD or MSE or both.

3.3 Another Method: Pure Regression Based Approach

The forecast for the data shown in 1 can be done using a pure regression based approach. In the regression based approach, observe that the $Y - variable$ (which is also called the dependent variable or response variable in regression) is the demand and the $X - variables$ of interest (also called the independent variables or explanatory variables) are the *week* and the *day*. The week is a numerical measure and increases from 1 to 2 to 3. Any variable that can be numerically ordered and if the ordering has some meaning, then that variable is called a numerical variable. In case of *week* the increasing order 1-2-3-4 indicates a temporal ordering such as week 2 comes after week 1 and week 3 comes after week 2 and so on (a time progression). Hence, the week variable can be treated as a numerical variable.

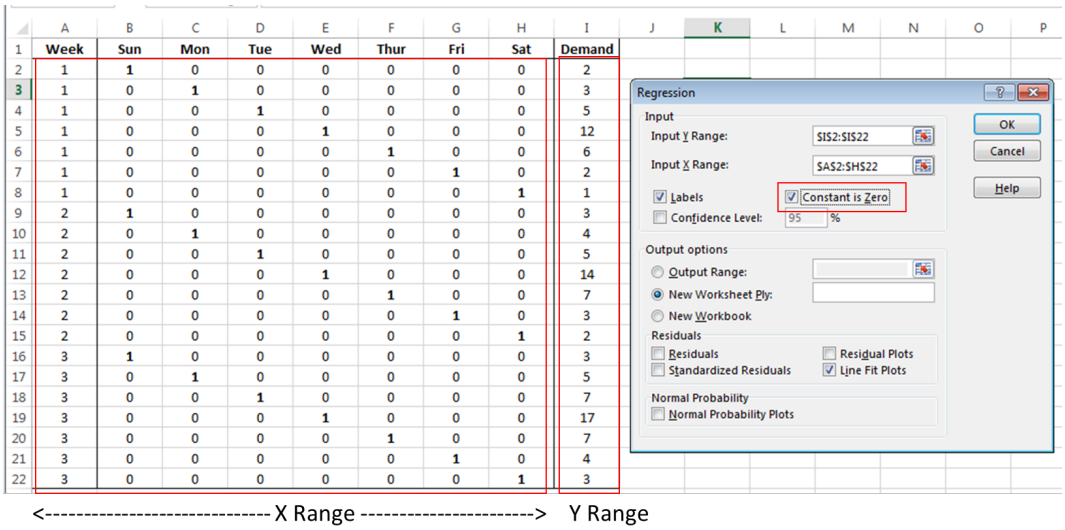
In case of the variable *day* no such ordering is possible. For example, we could have assigned numbers 1-7 for the week days. But the week days being repeating in nature we cannot say that day 7 necessarily comes after day 1. Hence, the variable *day* is not a numerical variable. Such variables in regression are called categorical variables. To handle such categorical variables in excel, it is necessary to create a separate variable for each category. For example, in this case we would need to create a variable for each day, i.e., Sun, Mon, Tue, Wed, Thur, Fri, and Sat. The variables are assigned a value of 1 if the day matches with the variable, else a value of zero is assigned. For example if the specific day is a Sunday, then we would assign a value of 1 to the created variable Sun, and assign 0 to all other created variables. This is shown in 12. We cannot work with period only since that would take away the effect of the day of the week completely and that would be incorrect since the demand depends significantly on the day of the week.

Figure 12: **Variable Creation for Categorical Variables for Regression**

Week	Sun	Mon	Tue	Wed	Thur	Fri	Sat	Demand
1	**1**	0	0	0	0	0	0	2
1	0	**1**	0	0	0	0	0	3
1	0	0	**1**	0	0	0	0	5
1	0	0	0	**1**	0	0	0	12
1	0	0	0	0	**1**	0	0	6
1	0	0	0	0	0	**1**	0	2
1	0	0	0	0	0	0	**1**	1
2	**1**	0	0	0	0	0	0	3
2	0	**1**	0	0	0	0	0	4
2	0	0	**1**	0	0	0	0	5
2	0	0	0	**1**	0	0	0	14
2	0	0	0	0	**1**	0	0	7
2	0	0	0	0	0	**1**	0	3
2	0	0	0	0	0	0	**1**	2
3	**1**	0	0	0	0	0	0	3
3	0	**1**	0	0	0	0	0	5
3	0	0	**1**	0	0	0	0	7
3	0	0	0	**1**	0	0	0	17
3	0	0	0	0	**1**	0	0	7
3	0	0	0	0	0	**1**	0	4
3	0	0	0	0	0	0	**1**	3
	Created Variables corresponding to each day of the week							

The regression estimation method is shown in 13. Note that the constant has been forced to zero by clicking the constant is zero. This is necessary while working with categorical variables whenever a variable is created corresponding to all the categories. There are other ways of doing the regression where the constant can be included along with categorical variables, however, for the purpose of this course we would do without the constant.

Figure 13: **Regression Model Fit with Categorical Variables**



Week	Sun	Mon	Tue	Wed	Thur	Fri	Sat	Demand
1	1	0	0	0	0	0	0	2
1	0	1	0	0	0	0	0	3
1	0	0	1	0	0	0	0	5
1	0	0	0	1	0	0	0	12
1	0	0	0	0	1	0	0	6
1	0	0	0	0	0	1	0	2
1	0	0	0	0	0	0	1	1
2	1	0	0	0	0	0	0	3
2	0	1	0	0	0	0	0	4
2	0	0	1	0	0	0	0	5
2	0	0	0	1	0	0	0	14
2	0	0	0	0	1	0	0	7
2	0	0	0	0	0	1	0	3
2	0	0	0	0	0	0	1	2
3	1	0	0	0	0	0	0	3
3	0	1	0	0	0	0	0	5
3	0	0	1	0	0	0	0	7
3	0	0	0	1	0	0	0	17
3	0	0	0	0	1	0	0	7
3	0	0	0	0	0	1	0	4
3	0	0	0	0	0	0	1	3

<-------------------------- X Range ----------------------> Y Range

The regression estimate is given in 14.

Figure 14: **Regression Estimate Results**

SUMMARY OUTPUT								
Regression Statistics								
Multiple R	0.996619035							
R Square	0.9932495							
Adjusted R Square	0.912691539							
Standard Error	0.712268213							
Observations	21							
ANOVA								
	df	*SS*	*MS*	*F*	*Significance F*			
Regression	8	970.4047619	121.3005952	239.0979242	4.85395E-12			
Residual	13	6.595238095	0.507326007					
Total	21	977						
	Coefficients	*Standard Error*	*t Stat*	*P-value*	*Lower 95%*	*Upper 95%*	*Lower 95.0%*	*Upper 95.0%*
Intercept	0	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A
Week	1.071428571	0.190361687	5.628383459	8.22285E-05	0.660177149	1.482679994	0.660177149	1.482679994
Sun	0.523809524	0.560409633	0.934690435	0.366985789	-0.686881882	1.73450093	-0.686881882	1.73450093
Mon	1.857142857	0.560409633	3.313902452	0.005596031	0.646451451	3.067834263	0.646451451	3.067834263
Tue	3.523809524	0.560409633	6.287917472	2.79774E-05	2.313118118	4.73450093	2.313118118	4.73450093
Wed	12.19047619	0.560409633	21.75279558	1.31264E-11	10.97978478	13.4011676	10.97978478	13.4011676
Thur	4.523809524	0.560409633	8.072326485	2.02605E-06	3.313118118	5.73450093	3.313118118	5.73450093
Fri	0.857142857	0.560409633	1.529493439	0.150104713	-0.353548549	2.067834263	-0.353548549	2.067834263
Sat	-0.142857143	0.560409633	-0.254915573	0.80277658	-1.353548549	1.067834263	-1.353548549	1.067834263

3.3.1 Forecasting With the Regression Model

The interpretation of the regression model is important. The coefficient of the numerical variable *week* can be interpreted as a weekly trend, i.e., the overall demand is increasing by 1.0714 units every week. The coefficients corresponding to the variables Sun, Mon, etc., (created from the categorical variable *day*) can be interpreted as the additive effect for that particular day. As an example, the forecast for the period corresponding to the Sunday of week W is $\hat{D}_{W,Sun} = 1.0714 \times W + 0.5238$. The coefficient estimate corresponding to Sun (0.5238) has been added to the base level week demand ($1.0714 \times W$). So the forecast demand for Sunday of Week 4 would be $1.0714 \times 4 + 0.5238 = 1.5952$. The complete forecast is shown in 15.

Figure 15: **Regression Estimate Forecast**

Week	Day	Regression Forecast
1	Sun	1.5952
1	Mon	2.9286
1	Tue	4.5952
1	Wed	13.2619
1	Thur	5.5952
1	Fri	1.9286
1	Sat	0.9286
2	Sun	2.6667
2	Mon	4.0000
2	Tue	5.6667
2	Wed	14.3333
2	Thur	6.6667
2	Fri	3.0000
2	Sat	2.0000
3	Sun	3.7381
3	Mon	5.0714
3	Tue	6.7381
3	Wed	15.4048
3	Thur	7.7381
3	Fri	4.0714
3	Sat	3.0714
4	**Sun**	**4.8095**
4	**Mon**	**6.1429**
4	**Tue**	**7.8095**
4	**Wed**	**16.4762**
4	**Thur**	**8.8095**
4	**Fri**	**5.1429**
4	**Sat**	**4.1429**

3.4 Choosing the Best Method

To choose between the regression approach and the time series approach, we would have to look at the forecast errors of the two methods. The comparison of the two methods is shown in table 16.

Figure 16: **Forecast Error Comparison of Time Series Based Method and Regression Model Based method**

Week	Day	Demand	Regression Forecast	MAD	MSE	Time Series Method	MAD	MSE
1	Sun	2	**1.5952**	0.4048	0.1638	**2.0854**	0.0854	0.0073
1	Mon	3	**2.9286**	0.0714	0.0051	**3.1287**	0.1287	0.0166
1	Tue	5	**4.5952**	0.4048	0.1638	**4.5202**	0.4798	0.2302
1	Wed	12	**13.2619**	1.2619	1.5924	**11.4952**	0.5048	0.2548
1	Thur	6	**5.5952**	0.4048	0.1638	**5.4466**	0.5534	0.3063
1	Fri	2	**1.9286**	0.0714	0.0051	**2.3833**	0.3833	0.1469
1	Sat	1	**0.9286**	0.0714	0.0051	**1.5638**	0.5638	0.3179
2	Sun	3	**2.6667**	0.3333	0.1111	**2.6745**	0.3255	0.1060
2	Mon	4	**4.0000**	0.0000	0.0000	**3.9782**	0.0218	0.0005
2	Tue	5	**5.6667**	0.6667	0.4444	**5.7018**	0.7018	0.4925
2	Wed	14	**14.3333**	0.3333	0.1111	**14.3918**	0.3918	0.1535
2	Thur	7	**6.6667**	0.3333	0.1111	**6.7713**	0.2287	0.0523
2	Fri	3	**3.0000**	0.0000	0.0000	**2.9435**	0.0565	0.0032
2	Sat	2	**2.0000**	0.0000	0.0000	**1.9195**	0.0805	0.0065
3	Sun	3	**3.7381**	0.7381	0.5448	**3.2636**	0.2636	0.0695
3	Mon	5	**5.0714**	0.0714	0.0051	**4.8278**	0.1722	0.0297
3	Tue	7	**6.7381**	0.2619	0.0686	**6.8834**	0.1166	0.0136
3	Wed	17	**15.4048**	1.5952	2.5448	**17.2885**	0.2885	0.0832
3	Thur	7	**7.7381**	0.7381	0.5448	**8.0961**	1.0961	1.2014
3	Fri	4	**4.0714**	0.0714	0.0051	**3.5037**	0.4963	0.2463
3	Sat	3	**3.0714**	0.0714	0.0051	**2.2751**	0.7249	0.5254
				0.3764	**0.3141**		**0.3649**	**0.2030**

While the performance of the two methods are very similar, the time series based method has lower MAD as well as MSE. Hence, the method of choice should be the time series based method.

3.5 Another Illustrative Example Discussed in Class

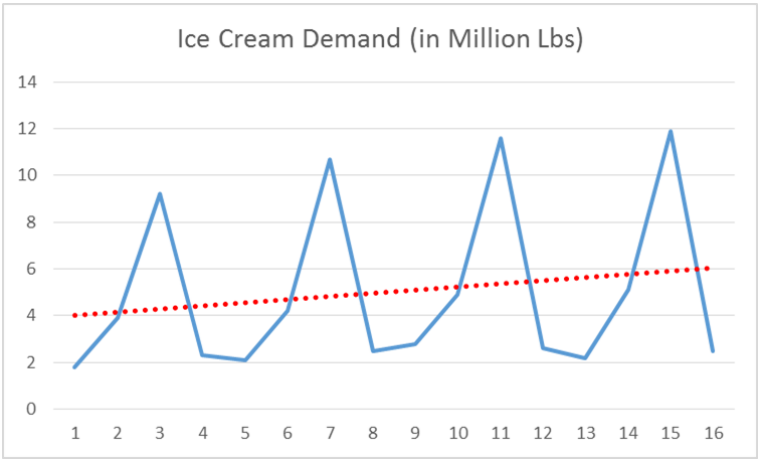
Consider the quarterly demand for ice creams in million pounds in the Mid Western United States for the past four years shown in table 17.

Figure 17: **Demand for Ice Creams (in Million lbs) in Mid Western United States**

Year	Quarter	Period	Demand
1	1	1	1.8
1	2	2	3.9
1	3	3	9.2
1	4	4	2.3
2	1	5	2.1
2	2	6	4.2
2	3	7	10.7
2	4	8	2.5
3	1	9	2.8
3	2	10	4.9
3	3	11	11.6
3	4	12	2.6
4	1	13	2.2
4	2	14	5.1
4	3	15	11.9
4	4	16	2.5

Clearly this data also has seasonality as shown in figure 18.

Figure 18: **Demand for Ice Creams (in Million lbs) in Mid Western United States**



3.5.1 Obtaining Deseasonalized Demand Data when Periodicity of Seasonality is Even Numbered

In this case the seasonality is yearly and since the demand is quarterly, the periodicity of seasonality of ice cream demand is 4. Since, the periodicity is an even number, unlike the previous example where the periodicity of seasonality was odd, there is no mid period, i.e., in this case the mid point of one cycle is between 2 and 3. To ensure that the period for which we want to get the deseasonalized demand is at the mid point of the periods considered for averaging, we would take $4 + 1 = 5$ periods. For example, if we want to get the deseasonalized demand for period 3, we would consider period 1 to 5. In doing so we would be distorting the estimate slightly since we are considering one addition period from the next cycle. Hence, we would give a weight of 1 to the start and the end period (period 1 and period 5) and give a weight of 2 to all other periods (2,3, and 4) and take the weighted average as the deseasonalized demand for period 3. So the deseasonalized demand for period 3 is $\bar{d}_3 = (d_1+2\times d_2+2\times d_3+2\times d_4+d_5)/8 = (1.8+2\times 3.9+2\times 9.2+2\times 2.3+2.1)/8 = 4.3375$. This is the only difference in this example as compared to the previous examples. All other steps are the same. The final forecast for the demand of ice cream in year 5 is shown in table 19.

Figure 19: **Forecast Demand for Ice Creams (in Million lbs) in Mid Western United States**

Year	Quarter	Period	Demand	Deseasonalized Demand	Fitted Deseasonalized Demand	Seasonality Factor	Forecast Demand
1	1	1	1.8		4.2606	0.4225	1.9331
1	2	2	3.9		4.3674	0.8930	3.9419
1	3	3	9.2	4.3375	4.4742	2.0562	9.4897
1	4	4	2.3	4.4125	4.5810	0.5021	2.1829
2	1	5	2.1	4.6375	4.6877	0.4480	2.1269
2	2	6	4.2	4.8500	4.7945	0.8760	4.3273
2	3	7	10.7	4.9625	4.9013	2.1831	10.3956
2	4	8	2.5	5.1375	5.0081	0.4992	2.3865
3	1	9	2.8	5.3375	5.1148	0.5474	2.3207
3	2	10	4.9	5.4625	5.2216	0.9384	4.7128
3	3	11	11.6	5.4000	5.3284	2.1770	11.3014
3	4	12	2.6	5.3500	5.4352	0.4784	2.5900
4	1	13	2.2	5.4125	5.5419	0.3970	2.5144
4	2	14	5.1	5.4375	5.6487	0.9029	5.0983
4	3	15	11.9		5.7555	2.0676	12.2073
4	4	16	2.5		5.8623	0.4265	2.7935
5	**1**	**17**			**5.9690**		**2.7082**
5	**2**	**18**			**6.0758**		**5.4838**
5	**3**	**19**			**6.1826**		**13.1132**
5	**4**	**20**			**6.2894**		**2.9970**

Practice Exercise: Try to estimate the forecast for year 6 also based on time series method shown above. Also, try to estimate the forecast based on regression method discussed earlier. Based on the forecast for the two methods, try to compute the MAD and MSE for both the forecasts. Which method is a more accurate method for this data?

3.6 Forecasting Without Seasonality in the Data

Whenever seasonality is not present in the data, time series averaging based methods work very well. The averaging based methods that are commonly used are the following:

3.6.1 Naive Forecast

In a Naive forecast method, the previous periods actual observed value is taken as the forecast for the next period. If D_t represents the actual observed data for period t and F_t represents the data for period t then the naive forecast can be stated as in equation 5.

$$\begin{aligned} \textbf{Naive Forecast: } & F_t = D_{t-1} \text{ For stable time series} \\ & F_t = D_{t-1} + (D_{t-1} - D_{t-2}) \text{ For Data with trend} \end{aligned} \tag{5}$$

Some advantages of naive forecast are (i) simplicity, (ii) ease of forecasting, (iii) good for stable environment and for the short run, and (iv) easy to understand. However, naive forecasts are not very accurate and are not useful for long term forecasts or forecasting in unstable situations.

3.6.2 Simple Moving Average

This is a technique that averages a number of recent actual values and updates as new values come. A simple n period moving average is given by equation 6.

$$F_t = \frac{1}{n} \sum_{i=1}^{n} D_{t-i} \tag{6}$$

The advantage of simple moving average is that it is responsive to changes in the data and random variations in the data. As the period increases the responsiveness to recent data goes down, but the forecast becomes more stable.

3.6.3 Weighted Moving Average

The weighted moving average is like the simple moving average where differential weights are associated with different period. Usually the more recent values are given more weights in computing the forecast. The n period weighted moving average is given by the equation 7

$$F_t = \sum_{i=1}^{n} w_i D_{t-i}$$
$$\sum_{i=1}^{n} w_i = 1 \tag{7}$$

As compared to Simple Moving Average, Weighted moving average is more responsive to recent changes in the data.

3.6.4 Exponential Smoothing

Exponential smoothing method is based on previous forecast plus a percentage of the forecast error. Exponential smoothing is given by equation 8.

$$F_t = F_{t-1} + \alpha(D_{t-1} - F_{t-1}) \tag{8}$$

In the above equation 8 α is called the smoothing constant. The term $D_{t-1} - F_{t-1}$ is the previous period's forecast error. Hence, α acts like a feedback term adjusting for previous period's forecast error.

4 Causal Forecasting Using Associative Models

Causal forecasting is done where the demand is relatively unstable and is influenced by environmental factors more than past demand. One illustrative example, is demand for construction materials is dependent on the overall state of the economy. Hence, past demand is not a good indicator of future demand for construction material such as concrete mix or steel. The future demand for construction material such as concrete mix and steel can be better predicted by using economic indicators such as Gross Domestic Product (GDP), Index of Industrial Production (IIP), Foreign Exchange Ratios (Forex Ratios) and Foreign Direct Investments (FDI). There may be many more factors such as interest rates, inflations rates, etc., that can influence the demand for construction material. Hence, to predict demand for construction material it is better to use associative / causal models using regression models or machine learning based data mining methods. We will introduce these forecasting methods through an illustrative example of bike demand from a bike ride sharing program.

4.1 Bike Demand Forecasting Example

The problem relates to demand forecasting for bike rides for a newly installed bike sharing program in a city. Below we describe the problem and the objective of building a forecasting model using the data related to past demand for bike sharing.

4.1.1 Problem Statement

Consider the data in the file *BikeDemandDaily.csv*. The data shows the demand for bike rides for a bike sharing program. There are two types of customer segments: (1) Casual customers who are not registered for the program but share bike ride on a spot payment basis, and (2) Registered customers who are registered for the program and share bike rides by paying in advance and have a member card.

4.1.2 Variable Description

The data in the file has the following information (variables):

1. **year**: Year index. There are two years in the sample.
2. **month**: Month index. Jan=1, Feb=2, ... , Dec=12.
3. **day**: Day index for a month and year.
4. **season**: Season index indicating four seasons.
5. **holiday**: Holiday index. 1 indicates holiday and 0 indicates not a holiday.
6. **workingday**: Working day index. 1 indicates working day and 0 indicates not a working day.
7. **meanatemp**: Average daily temperature (degree Celsius).
8. **maxatemp**: Maximum daily temperature (degree Celsius).
9. **minatemp**: Minimum daily temperature (degree Celsius).
10. **sdatemp**: Standard deviation of the hourly temperature during a day.
11. **meanhumidity**: Average humidity for a day (percentage).
12. **maxhumidity**: Maximum humidity for a day (percentage).
13. **minhumidity**: Minimum humidity for a day (percentage).
14. **sdhumidity**: Standard deviation of humidity for a day.
15. **meanwindspeed**: Average wind speed in kmph.
16. **maxwindspeed**: Maximum wind speed in kmph.
17. **minwindspeed**: Minimum wind speed in kmph.
18. **sdwindspeed**: Standard deviation of wind speed.
19. **Casual**: Number of casual customers using the bike during the day.
20. **Registered**: Number of registered customers using the bike sharing program.
21. **Total**: Total number of daily customers.

4.1.3 Objective of Analysis

The following list illustrates some of the expected analysis outcome:

1. Understand the pattern of bike demand for causal, registered and total number of bikes demanded.
2. Plot relevant graphs to understand the demand pattern.
3. Plot variable graphs to understand important predictors.
4. Visualization of data.
5. Use moving average method for forecasting.
6. Use regression analysis for forecasting.
7. Use regression to decide which are the important variables.
8. Use some machine learning methods for better prediction.
9. Management report preparation and decision framework analysis.
10. Simulate data to understand loss in profit at various levels of inventory of bikes.

4.2 Understanding Predictive Accuracy

4.2.1 Constructing Train and Test Sample

In predictive analytics, computing model accuracy is critical. However, it is not advisable to use the same data sample that is used to build a predictive model to also estimate model accuracy. This is because often models tend to perform well when tested on the same data sample that has been used to construct a model. However, the real utility of a predictive model is when the model is able to predict / forecast for data samples that are not included in the model estimation. Predictive accuracy measured on a sample that has not been used to compute the model parameters is often called the *Out of Sample Prediction Accuracy*.

One strategy for computing out of sample prediction accuracy of a model is to split a data sample into two randomly selected subsamples, called the *Train Sample* and the *Test Sample*. The train sample is used to construct a model and estimate model parameters and the test sample is used to construct a measure for prediction accuracy for a model. Often, this process can be repeated to get the prediction accuracy over a number of test and train samples. This can be used to construct a confidence interval of the prediction accuracy measure. All these would be illustrated in relevant sections in the document.

First, let us see how to construct a train and test sample in R. First, we would need to decide the size of the train and test sample. While there is no scientific guideline on the split ratio, a 80% train sample and a 20% test sample (80:20 split of the data) is often used for practical purposes. Other split ratios such as 90:10 or 70:30 or 60:40 can be used. As the size of the train sample decreases, the stability of the model parameter estimates decreases. A more robust model is one which can achieve a relatively higher stability of model parameter estimates even when the train sample size decreases. We will discuss about model stability further in a future section. For now, we would be using a 80:20 split of the data sample into train and test samples.

In R we would use the command **sample.int** to create a sample of indices for the train and test sample. The first argument of sample **sample.int** indicates the integer range to select from. In our example, the integer range to be sampled from is the total number of rows in the data sample. Second argument *size* specifies the size of the randomly selected sample. The R code below illustrates how we can split the data sample into train and test sample.

```
> d<-read.csv("BikeDemandDaily.csv", header=TRUE)
> n<-dim(d)[1];
> n
[1] 456
> ind<-sample.int(n, size=150)
> dtrain<-d[-ind,];
> dtest<-d[ind,]
```

The total number of observations is *n=456*. We have created a random sample of 150 indices of row numbers using the *sample.int()* function. The command *d[ind,]* selects from the data table *d* those rows whose numbers are in the randomly generated sample of indices *ind*. We call this sample the *Test* sample whose size is 150 observations. The command *d[-ind,]* returns the all rows from the datat table *d* other than those rows whose indices are in the ramdom sample *ind*. We call this the *Train* sample and has a size of 456-150=306 observations. We would use the train sample to extimate our prediction models and we would use the test sample to test the forecasting / prediction accuracy of the estimated models.

4.2.2 Constructing the Root Mean Square Prediction Error (RMSPE)

Having constructed the *Train* and *Test* samples, we would need to construct a prediction error measure and function. Similar to the Mean Squared Error (MSE) that we have seen in section [3.2], we would use a Root Mean Squared Prediction Error (RMSPE) function for measuring the prediction accuracy of a model. The RMSPE measure is defined as the square root of the mean square of

the difference between the actual demand realizations and the predicted demand by a model corresponding to the observations in the *Test* data sample. This is shown in Equation 9.

$$RMSPE = \sqrt{\frac{1}{n_{Test}} \sum_{i=1}^{n_{Test}} (y_{Test,i} - \hat{y}_{pred,i})} \tag{9}$$

Since, this measure would be used repeatedly throughout the analysis for measuring prediction accuracy of different models, it is advisable to create a function in R which can then be used repeatedly like any other R functions.

4.2.3 User Defined Functions in R

User defined functions in R are a set if codes that are grouped together under a common name that can be referenced to execute the codes within a function. Often function take inputs which are passed on to a function as *arguments*. Also, a function often returns the output of the function through a return statement. All variables and objects within a function are local to a function and cannot be generally accessed from outside the function. The only interaction that a function has with the outside environment is through the input arguments and the return value. The general structure of a R function is shown below:

```
myfunction <- function(arg1, arg2, ... ){
statements
return(object)
}
```

As an illustrative example, we can write a function to compute the area of a circle as shown below.

```
> CircArea<-function(radius){
+ area=pi*radius^2;
+ return(area);
+ }
> CircArea(3)
[1] 28.27433
> CircArea(4)
[1] 50.26548
> CircArea(0.5)
[1] 0.7853982
> CircArea(21.845)
[1] 1499.181
```

As we can see from the example above, the function *CircArea* takes the *radius* as the only argument and returns the area of the circle corresponding to the

given radius. The function *CircArea()* can now be used repeatedly to compute areas of different circles. It is possible to pass more than one argument to a function.

4.2.4 RMSPE Function

We define the RMSPE function to measure prediction accuracy of models.

```
> RMSPE=function(d,p){
+ y1=mean((d-p)^2);
+ y2=sqrt(y1)
+ return(y2);
+ }
```

As an illustrative example, if observed demand is {*3,9,5,7,3,7,10,3,4,7*} and predicted demand is {*4,8,3,6,8,12,3,5,4,9*} for a particular product for 10 consecutive months in a year, the prediction error can be computed using the *RMSPE* function as:

```
> demand<-c(3,9,5,7,3,7,10,3,4,7)
> predicted<-c(4,8,3,6,8,12,3,5,4,9)
> RMSPE(demand, predicted)
[1] 3.376389
```

4.3 Building the Bike Forecast Model

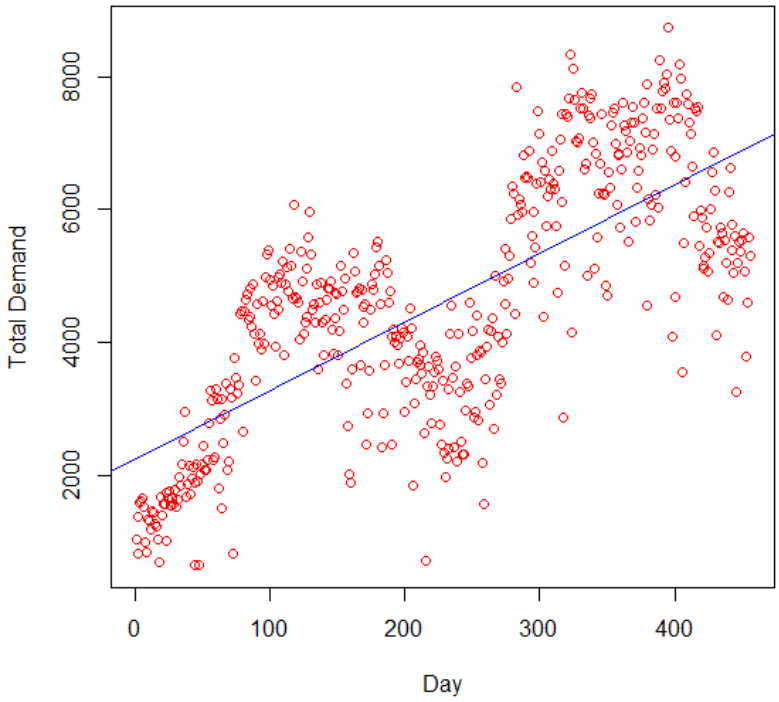
Below I describe the steps in building a good demand forecasting model.

4.3.1 Visualization of the Demand Data

The first step to building a model is to visualize the data. We will first visualize the daily bike demand pattern over time. To visualize the bike demand pattern, we would use a *scatterplot* of the data. Figure 20 shows the scatterplot corresponding to the complete sample of observations of total demand.

```
> plot(d$Total, col=2, xlab="Day", ylab="Total Demand")
> abline(lm(d$Total~d$Index), col=4)
```

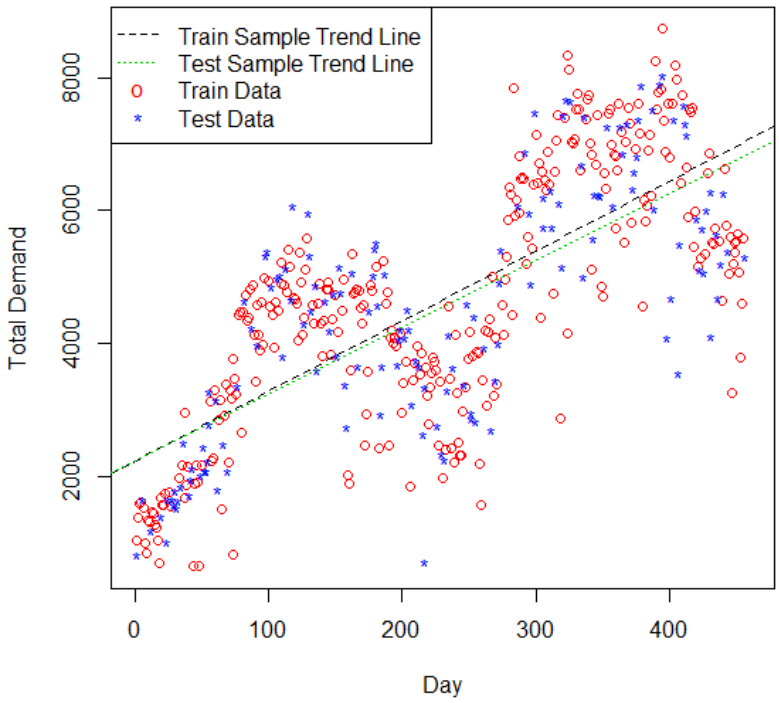
Figure 20: **Scatterplot of Daily Bike Demand (Complete Sample of Observations)**



To check if the split of the data into train and test samples has been random, we plot the train and the test sample separately on the same graph as shown in Figure 21.

```
> plot(dtrain$Index, dtrain$Total, xlim=c(1,460),
+ xlab="Day", ylab="Total Demand", col=2)
> abline(lm(dtrain$Total~dtrain$Index), lty=2, col=1)
> points(dtest$Index, dtest$Total, col=4, pch="*");
> abline(lm(dtest$Total~dtest$Index), lty=3, col=3);
> legend("topleft", legend=c("Train Sample Trend Line",
+ "Test Sample Trend Line", "Train Data", "Test Data"),
+ lty=c(2,3,NA,NA), col=c(1,3,2,4), pch=c(NA,NA,"o","*"))
```

Figure 21: **Scatterplot of Daily Bike Demand (Train and Test Samples)**



Visually it seems from Figure 21 that both train and test data are fair representation of the complete sample and the split have been fairly random. Later, we would discuss about non-random split of the data into train and test sample.

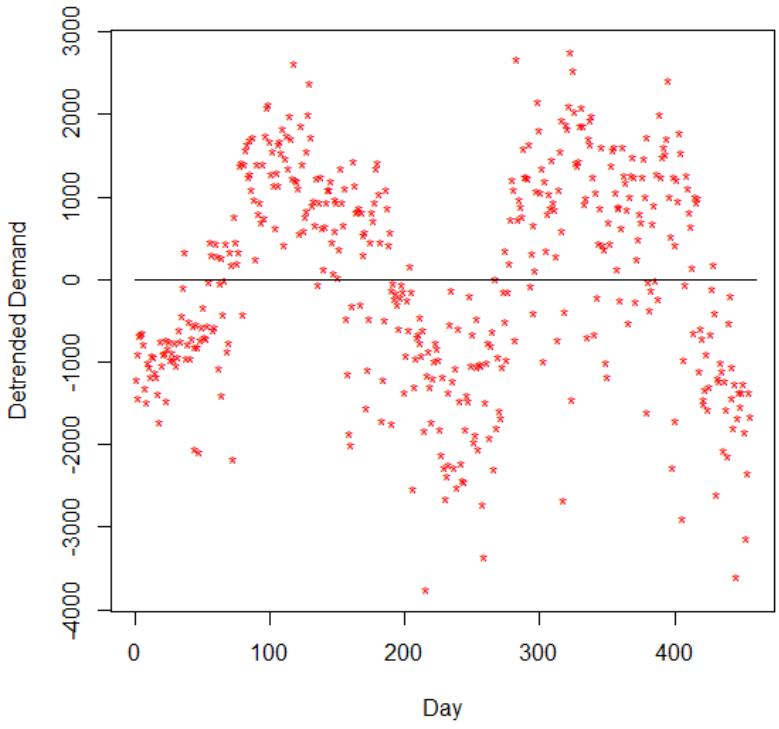
From the data the following can be inferred.

1. The demand pattern clearly shows seasonality of demand.
2. Overall the demand is showing an increasing trend.

To visualize seasonality better, it is necessary to detrend the data and plot the detrended demand data. Demand can be detrended by substracting the trend component from the demand data. The trend component is estimated by fitting a trend line through a linear regression model and subtracting the predicted values from the trend line regression from the demand data. The process of detrending the demand data and plotting the detrended demand data is shown in the following block of R code and Figure 22.

```
> m1<-lm(Total~Index, data=d)
> p1<-predict(m1)
> detrendedDemand<-d$Total-p1
> plot(d$Index, detrendedDemand, xlab="Day",
+ ylab="Detrended Demand", col=2, pch="*")
> lines(c(0,460),c(0,0))
```

Figure 22: **Scatterplot of Daily Bike Demand (Detrended)**



From the detrended demand plot in Figure 22 the seasonal nature of the data is clearly evident.

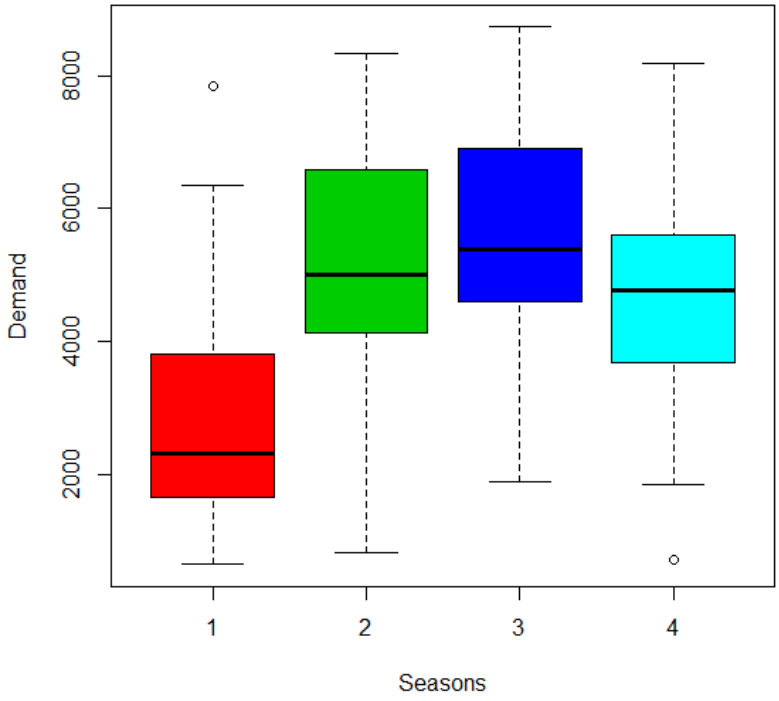
4.3.2 Some More Visualization of Data to Understand Explanatory Variable Associations

It is often helpful to visualize variable relationships through graphical means. Specifically, it is helpful to graphically understand the relationship between

some of the explanatory variables that may be associated with the demand (independent variable) and the independent variable. First, we look at a box plot of demand on the variable season. Figure 23 shows that there is clearly significant difference in demand between the four seasons in the data.

```
> boxplot(d$Total~d$season, xlab="Seasons",
+ ylab="Demand", col=c(2,3,4,5))
```

Figure 23: **Box Plot of Bike Demand on Seasons**

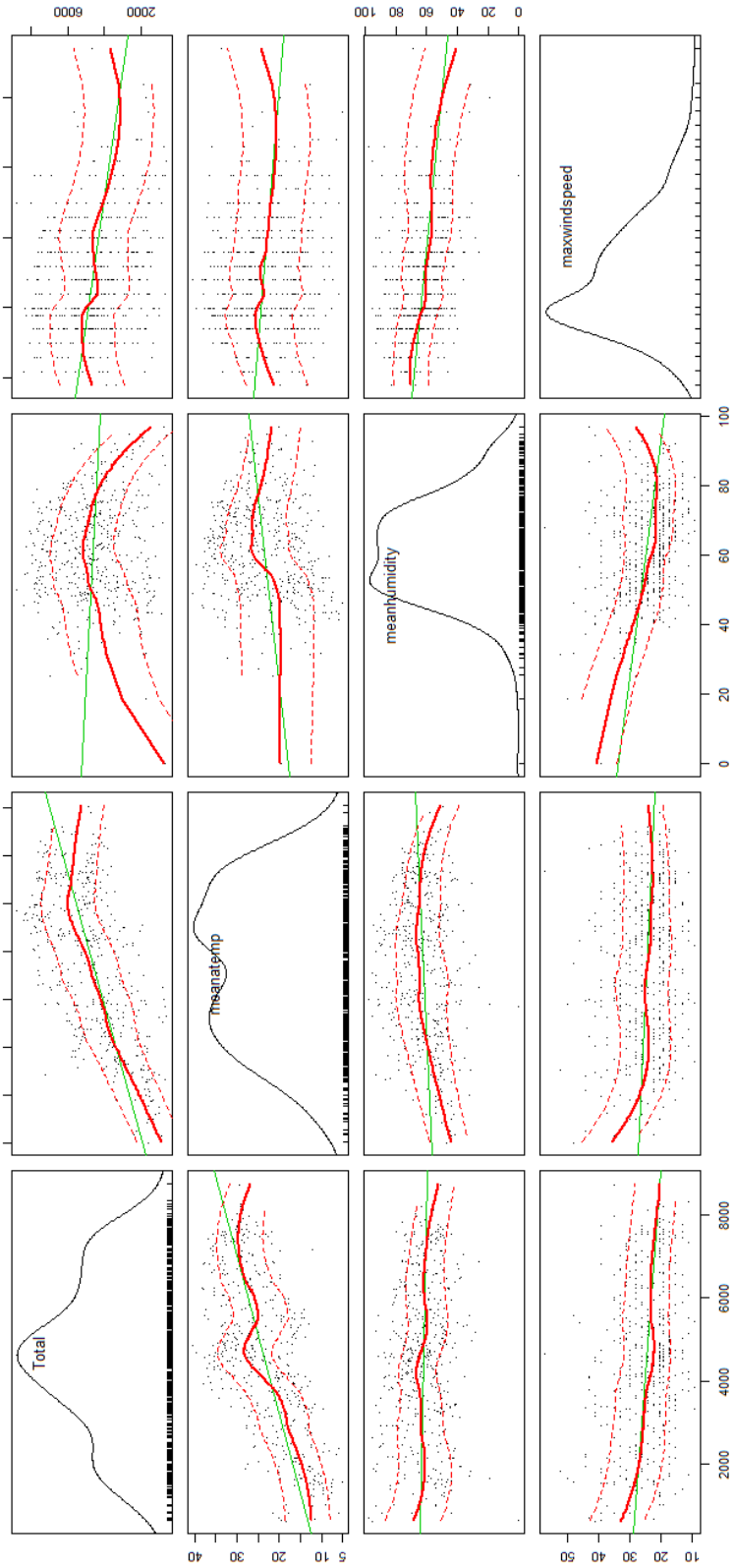


Similarly, let us examine the relationships between bike demand and the variables *meanatemp*, *meanhumidity*, and *maxwindspeed*. Figure 24 shows the relationships between temperature, humidity, wind speed, and demand.

```
> library(car)
> library(MASS)
```

```
> scatterplotMatrix(~Total+meanatemp
+meanhumidity+maxwindspeed, data=d, pch=".")
```

Figure 24: **Scatter Plot Matrix of Demand, Temperature, Humidity and Wind Speed**



The following can be inferred from the scatter plot matrix (Refer to the three plots on the first row).

1. As temperature increases demand increases. However, when temperature is in the higher ranges, demand goes decreases with increase in temperature.
2. The relationship between humidity and demand is like and inverted U shape.
3. Demand is negatively correlated with wind-speed.

The plots can be individually generated using the *plot()* command without using *scatterplotMatrix()* command.

4.4 Fitting a Linear Regression Model

A linear regression model is statistical model which estimates a functional relationship between the independent variable and one or more dependent variables using a linear combination of the explanatory variables or some transformation of the explanatory variables. A linear regression also assumes a specific parametric distribution of the errors associated with each observation.

Equation 10 specifies a standard specification of a linear regression model.

$$y_i = \beta_0 + \beta_1 x1_i + \cdots + \beta_k xk_i + \epsilon_i;\ \epsilon_i \sim N(0, \sigma^2) \tag{10}$$

In the Equation 10 $N(0, \sigma^2)$ denotes a normal distribution with 0 mean and σ^2 variance. The task of fitting a linear regression on a data sample is to estimate the model parameters $\{\beta_0, \beta_1, \cdots, \beta_k\}$ from the data such that some function of the errors is minimized. Usually, simple linear regression or multiple linear regression estimation is done by minimizing the sum of the squares of the errors. In the regression model of 10 the parameter β_0 is called the *intercept* and the other parameters are called the *slope* parameters corresponding to each of the independent variables.

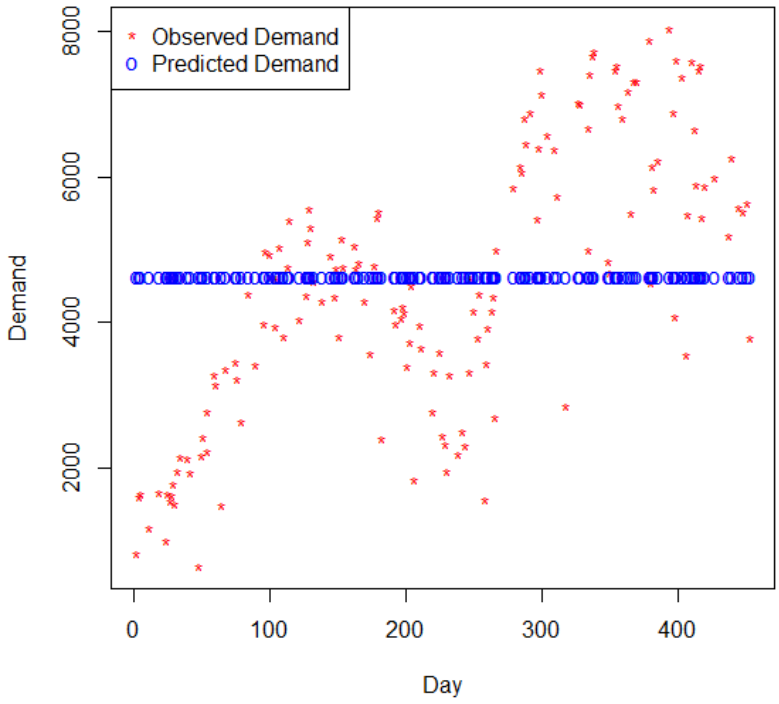
For the bike demand estimation problem we would use the train sample to estimate the models and use the test set to compute model accuracy of prediction. The simplest model that can be fitted is a model with only the intercept and no explanatory / independent variables. We will start with the simplest model and then gradually we would start including independent variables.

4.4.1 Linear Model with Only Intercept

The following code block shows fitting a linear model with only intercept. We also plot the predicted values against the actual demand from the test sample (Figure 25).

```
> m0<-lm(Total~1, data=dtrain)
> p0<-predict(m0, newdata=dtest)
> plot(dtest$Index, dtest$Total, pch="*",
+ col=2, xlab="Day", ylab="Demand");
> points(dtest$Index, p0, pch="o", col=4)
> legend("topleft",
+ legend=c("Observed Demand", "Predicted Demand"),
+ pch=c("*","o"), col=c(2,4))
```

Figure 25: **Actual versus Predicted Demand Plot for Intercept Only Model**



It can be shown that the estimate for intercept parameter for the intercept only model is the average demand for the train sample.

```
> summary(m0)
```

```
Call:
lm(formula = Total ~ 1, data = dtrain)

Residuals:
    Min      1Q  Median      3Q     Max
-3998.0 -1265.5    15.5  1437.5  4103.0

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   4633.0      107.1   43.24   <2e-16 ***
---
Signif. codes:  0    ***    0.001    **    0.01
    *    0.05    .    0.1          1

Residual standard error: 1874 on 305 degrees of freedom

> mean(dtrain$Total)
[1] 4632.974
```

The RMSPE measure of prediction error is 1856.975 for the intercept only model.

```
> RMSPE(dtest$Total, p0)
[1] 1856.975
```

4.4.2 Intercept and Trend Model

The intercept only model surely does not provide us with a reasonably accurate prediction of demand. This is because the demand is not stable and has significant seasonality as well as increasing trend. Surely, adding trend to the prediction model is likely to improve the prediction accuracy and decrease the RMSPE measure of prediction error. Below we show the model fitting with intercept as well as a trend line included in the linear regression model. Figure 26 plot of predicted versus observed demand.

```
> m1<-lm(Total~Index, data=dtrain)
> summary(m1)

Call:
lm(formula = Total ~ Index, data = dtrain)

Residuals:
   Min      1Q  Median      3Q     Max
```

```
 -3782  -1010      78    1066    2728

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 2251.2446   144.6803   15.56   <2e-16 ***
Index         10.3853     0.5459   19.02   <2e-16 ***
---
Signif. codes:  0    ***    0.001    **    0.01
    *    0.05    .    0.1         1

Residual standard error: 1268 on 304 degrees of freedom
Multiple R-squared:  0.5435,    Adjusted R-squared:  0.542
F-statistic: 361.9 on 1 and 304 DF,  p-value: < 2.2e-16

> p1<-predict(m1, newdata=dtest)
> plot(dtest$Index, dtest$Total, pch="*",
+ col=2, xlab="Day", ylab="Demand");
> points(dtest$Index, p1, pch="o", col=4)
> legend("topleft",
+ legend=c("Observed Demand", "Predicted Demand"),
+ pch=c("*","o"), col=c(2,4))
```

It can be seen from 26 that the demand forecast captures the increasing trend. The intercept estimate for trend line in model fit summary is positive and significant. This shows that the trend line is important for forecasting. Also, the error of prediction has decreased from 1856.975 (RMSPE value) to 1286.959. This is a significant improvement.

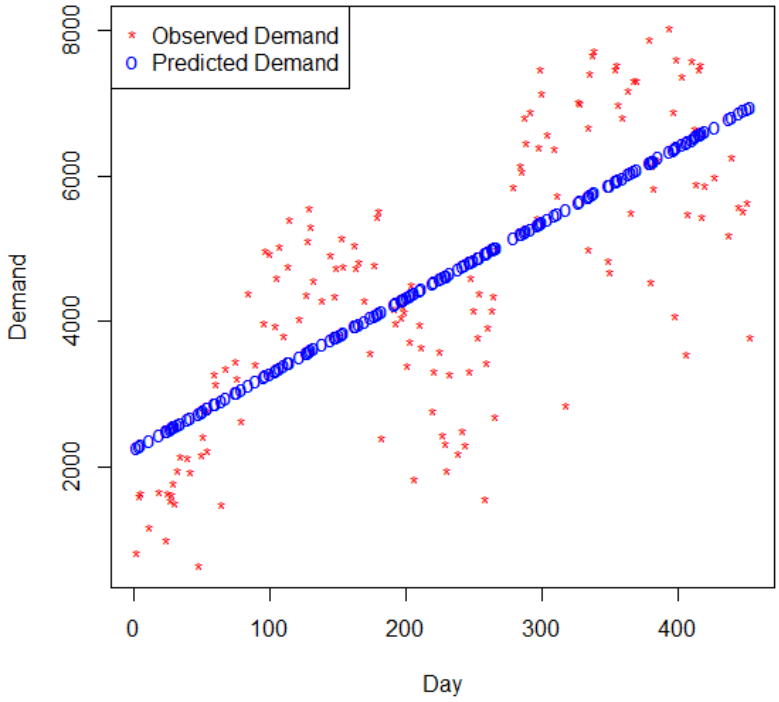
```
> RMSPE(dtest$Total, p1)
[1] 1286.959
```

Also, the two models can be compared using an ANOVA estimate as shown. The significant value of the F-statistic shows that model with trend ($m1$) is a better demand forecast model as compared to the intercept only model ($m0$).

```
> anova(m0,m1)
Analysis of Variance Table

Model 1: Total ~ 1
Model 2: Total ~ Index
  Res.Df        RSS Df Sum of Sq     F     Pr(>F)
1    305 1071405162
2    304  489119918  1 582285244 361.9 < 2.2e-16 ***
---
Signif. codes:  0    ***    0.001    **    0.01
```

```
*      0.05      .      0.1            1
```

Figure 26: **Actual versus Predicted Demand Plot for Intercept and Trend Only Model**



4.4.3 Adding Seasonality to Linear Model

Seasonality is represented by the categorical variable *season* which can be included in the linear model to capture the effect of seasonality in addition to the trend line in model $m1$.

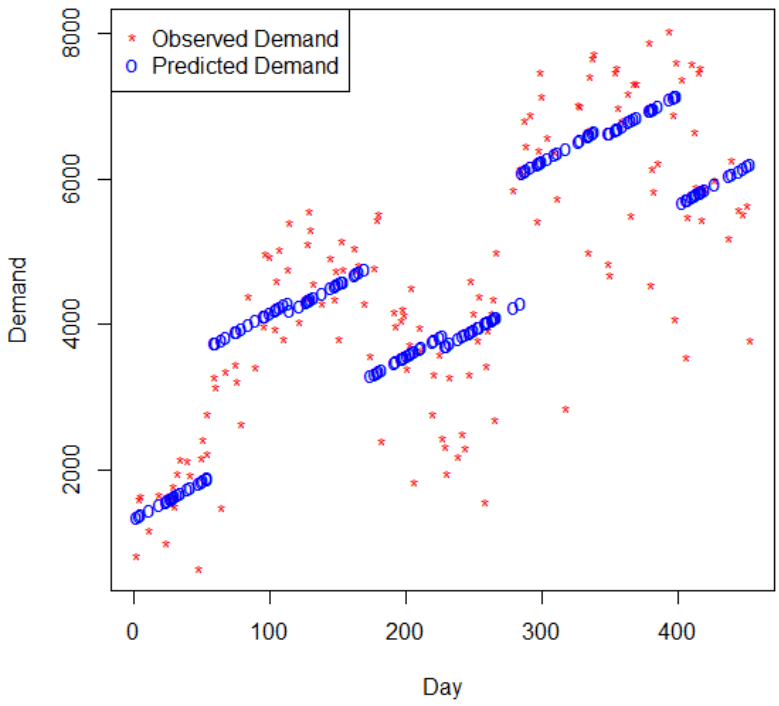
```
> m2<-lm(Total~Index+as.factor(season), data=dtrain);
> summary(m2)

Call:
lm(formula = Total ~ Index + as.factor(season),
+ data = dtrain)
```

```
Residuals:
    Min      1Q  Median      3Q     Max
-3064.1  -565.3    21.8   546.5  3569.3

Coefficients:
                     Estimate Std. Error t value Pr(>|t|)
(Intercept)        1328.1394   131.0440  10.135   <2e-16 ***
Index                10.4082     0.4754  21.892   <2e-16 ***
as.factor(season)2 1794.1732   158.3111  11.333   <2e-16 ***
as.factor(season)3 1670.4589   165.4478  10.097   <2e-16 ***
as.factor(season)4  159.6668   177.5965   0.899    0.369
---
Signif. codes:  0    ***    0.001    **    0.01
    *    0.05    .    0.1         1

Residual standard error: 962.8 on 301 degrees of freedom
Multiple R-squared:  0.7396,    Adjusted R-squared:  0.7361
F-statistic: 213.7 on 4 and 301 DF,  p-value: < 2.2e-16

> p2<-predict(m2, newdata=dtest)
> plot(dtest$Index, dtest$Total, pch="*",
+ col=2, xlab="Day", ylab="Demand");
> points(dtest$Index, p2, pch="o", col=4)
> legend("topleft",
+ legend=c("Observed Demand", "Predicted Demand"),
+ pch=c("*","o"), col=c(2,4))
```

From the plot in Figure 27 it is evident that the predicted model includes corrections for both trend and seasonality and this has improved the prediction accuracy significantly as is evident from the reduction in prediction error RMSPE to 1001.071.

```
> RMSPE(dtest$Total, p2)
[1] 1001.071
```

Figure 27: **Actual versus Predicted Demand Plot for Trend and Seasonality Corrected Model**



4.4.4 Adding Other Explanatory Variables

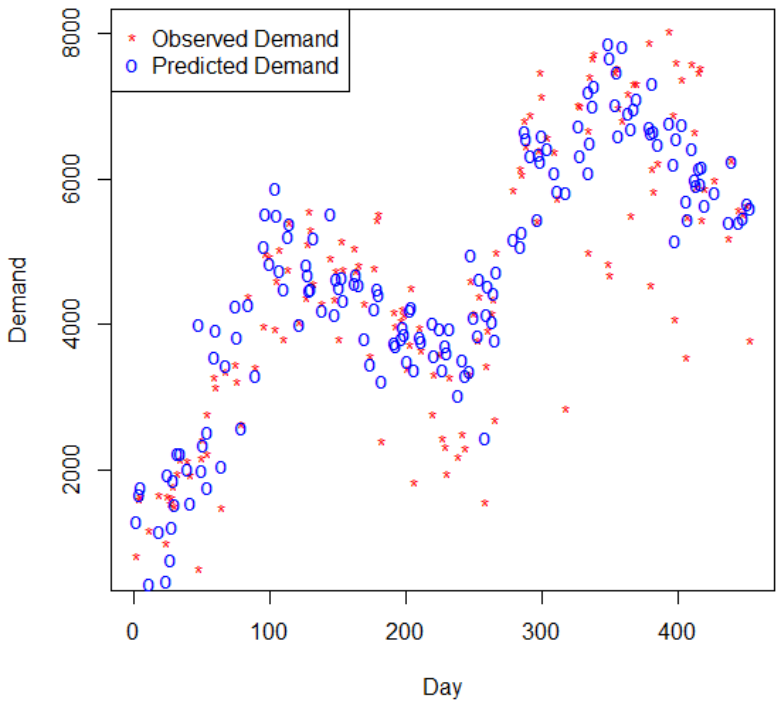
Adding other explanatory variables such as temperature, humidity, wind speed, working day, month, and holidays are likely to improve the demand forecast, since demand for bike rides surely depend significantly on the environmental variables. Let us see the effect of adding the remaining dependent variables on the accuracy of prediction of the linear model.

```
> m3<-lm(Total~Index+as.factor(season)+as.factor(holiday)
+meanatemp+meanwindspeed+meanhumidity, data=dtrain);
> summary(m3)

Call:
lm(formula = Total ~ Index + as.factor(season)
+ as.factor(holiday) +  meanatemp
```

```
+ meanwindspeed + meanhumidity,
data = dtrain)

Residuals:
     Min       1Q   Median       3Q      Max
-2065.50  -423.54    11.41   401.79  2834.47

Coefficients:
                    Estimate Std. Error t value Pr(>|t|)
(Intercept)         2349.063    300.266   7.823 9.08e-14 ***
Index                  9.192      0.362  25.394  < 2e-16 ***
as.factor(season)2   688.453    152.823   4.505 9.56e-06 ***
as.factor(season)3   -27.102    193.321  -0.140    0.889
as.factor(season)4    -1.378    139.078  -0.010    0.992
as.factor(holiday)1   82.771    243.236   0.340    0.734
meanatemp            109.858      9.116  12.052  < 2e-16 ***
meanwindspeed        -51.291      8.736  -5.871 1.16e-08 ***
meanhumidity         -30.901      3.120  -9.904  < 2e-16 ***
---
Signif. codes:  0    ***    0.001    **    0.01
    *    0.05    .    0.1         1

Residual standard error: 714.5 on 297 degrees of freedom
Multiple R-squared:  0.8585,    Adjusted R-squared:  0.8547
F-statistic: 225.2 on 8 and 297 DF,  p-value: < 2.2e-16

> p3<-predict(m3, newdata=dtest)
> plot(dtest$Index, dtest$Total, pch="*",
+ col=2, xlab="Day", ylab="Demand");
> points(dtest$Index, p3, pch="o", col=4)
> legend("topleft",
+ legend=c("Observed Demand", "Predicted Demand"),
+ pch=c("*","o"), col=c(2,4))
>
> RMSPE(dtest$Total, p3)
[1] 864.2869
```

We can observe that addition of other dependent variables, representing factors that drive demand for bike sharing, significantly improve the prediction accuracy of the forecasting model. The RMSPE value is reduced to 864.2869. Also, the plot of the predicted values of demand against actual demand shows that the prediction model is able to capture the trend in demand much better than any of the previous models (Refer to Figure 29.

Figure 28: **Actual versus Predicted Demand Plot for Complete Model**



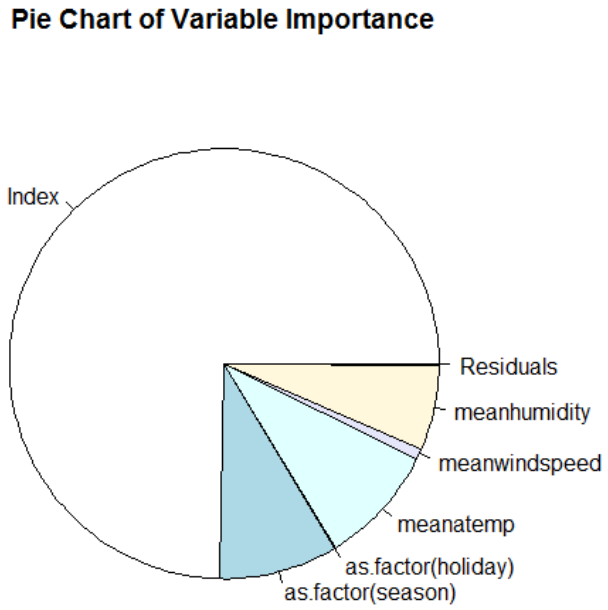
Anova model can be used to understand the contribution of each individual variable in explaining the variance of the dependent variable.

```
> a<-anova(m3)
> a
Analysis of Variance Table

Response: Total
                     Df     Sum Sq    Mean Sq  F value     Pr(>F)
Index                 1  582285244  582285244 1140.5640 < 2.2e-16 ***
as.factor(season)     3  210115051   70038350  137.1892 < 2.2e-16 ***
as.factor(holiday)    1     287263     287263    0.5627 0.4537752
meanatemp             1   71069776   71069776  139.2095 < 2.2e-16 ***
meanwindspeed         1    5948724    5948724   11.6522 0.0007304 ***
meanhumidity          1   50073485   50073485   98.0825 < 2.2e-16 ***
Residuals           297  151625619     510524
```

```
---
Signif. codes:  0    ***    0.001    **    0.01
+    *    0.05    .    0.1          1
> pie(a[,3], labels=row.names(a),
+ main="Pie Chart of Variable Importance")
```

It can be observed that the trend (Index) explains most of the variance followed by season, temperature and humidity. This can also be visually observed from the pie chart plot in Figure **??**.

Figure 29: **Pie Chart of Variable Importance from Anova**



4.5 Generalized Linear Models: Gaussian, Poisson and Negative Binomial Models

Ordinary linear regression depends on several assumptions such as uncorrelated error and dependent variables, normality of error distribution and constant error

variance. However, often in real datasets, many of these assumptions are not satisfied. If any of these assumptions on which consistent estimation of ordinary linear regression depend is not satisfied, some correction is required for consistent estimation of model parameters. Generalized Linear Models (GLMs) are a class of models that are flexible generalizations of the ordinary linear models. GLMs can be used for several different error distributions such as generalized Gaussian with non-constant variance, Poisson distribution and negative binomial distribution which are used for modeling count variables. The demand for bike is a discrete count variable and hence is likely to be better modeled using a Generalized Linear Model (GLM) using either a Poisson or a Negative Binomial distribution. Below, we show the model estimation using Gaussian, Poisson, and Negative Binomial distribution. GLMs are fitted using the *glm()* function. The error distribution is specified in the parameter *family* within *glm()* function. For example, for using a *Gaussian* distribution, the *family* parameter has to be set at *gaussian* (*family="gaussian"*). The subsequent codes are self explanatory.

4.5.1 GLM with Gaussian Distribution

```
> m4<-glm(Total~Index+as.factor(season)+as.factor(holiday)
+meanatemp+meanwindspeed+meanhumidity,
  data=dtrain, family="gaussian");
> summary(m4)

Call:
glm(formula = Total ~ Index + as.factor(season) +
as.factor(holiday) +  meanatemp + meanwindspeed
+ meanhumidity, family = "gaussian",  data = dtrain)

Deviance Residuals:
     Min        1Q    Median        3Q       Max
-2065.50   -423.54     11.41    401.79   2834.47

Coefficients:
                    Estimate Std. Error t value Pr(>|t|)
(Intercept)         2349.063    300.266   7.823 9.08e-14 ***
Index                  9.192      0.362  25.394  < 2e-16 ***
as.factor(season)2   688.453    152.823   4.505 9.56e-06 ***
as.factor(season)3   -27.102    193.321  -0.140    0.889
as.factor(season)4    -1.378    139.078  -0.010    0.992
as.factor(holiday)1   82.771    243.236   0.340    0.734
meanatemp            109.858      9.116  12.052  < 2e-16 ***
meanwindspeed        -51.291      8.736  -5.871 1.16e-08 ***
meanhumidity         -30.901      3.120  -9.904  < 2e-16 ***
---
Signif. codes:  0    ***    0.001    **    0.01    *    0.05    .    0.1
```

```
(Dispersion parameter for gaussian family taken to be 510524)

    Null deviance: 1071405162  on 305  degrees of freedom
Residual deviance:  151625619  on 297  degrees of freedom
AIC: 4901.1

Number of Fisher Scoring iterations: 2

> p4<-predict(m3, newdata=dtest)
>
> RMSPE(dtest$Total, p4);
[1] 864.2869
```

4.5.2 GLM with Poisson Distribution

```
> m5<-glm(Total~Index+as.factor(season)+as.factor(holiday)
+meanatemp+meanwindspeed+meanhumidity,
  data=dtrain, family="poisson");
> summary(m5)

Call:
glm(formula = Total ~ Index + as.factor(season) +
  as.factor(holiday) + meanatemp + meanwindspeed
  + meanhumidity, family = "poisson", data = dtrain)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-48.671   -7.340    0.080    6.664   45.194

Coefficients:
                     Estimate Std. Error  z value Pr(>|z|)
(Intercept)         7.703e+00  6.864e-03 1122.211  < 2e-16 ***
Index               2.058e-03  7.662e-06  268.618  < 2e-16 ***
as.factor(season)2  1.956e-01  3.416e-03   57.248  < 2e-16 ***
as.factor(season)3  1.303e-02  4.186e-03    3.113  0.00185 **
as.factor(season)4  9.389e-02  3.137e-03   29.933  < 2e-16 ***
as.factor(holiday)1 -4.525e-03  4.828e-03   -0.937  0.34863
meanatemp           2.774e-02  1.984e-04  139.818  < 2e-16 ***
meanwindspeed      -1.028e-02  1.925e-04  -53.394  < 2e-16 ***
meanhumidity       -6.610e-03  6.736e-05  -98.133  < 2e-16 ***
---
Signif. codes:  0    ***    0.001    **    0.01    *    0.05    .    0.1

(Dispersion parameter for poisson family taken to be 1)
```

```
    Null deviance: 252905  on 305  degrees of freedom
Residual deviance:  42946  on 297  degrees of freedom
AIC: 46076

Number of Fisher Scoring iterations: 4

> p5<-predict(m5, newdata=dtest, type="response")
>
> RMSPE(dtest$Total, p5);
[1] 929.1312
```

4.5.3 GLM with Negative Binomial Distribution

```
> m6<-glm.nb(Total~Index+as.factor(season)
+as.factor(holiday)+meanatemp+meanwindspeed
+meanhumidity, data=dtrain);
> summary(m6)

Call:
glm.nb(formula = Total ~ Index + as.factor(season)
 + as.factor(holiday) +  meanatemp + meanwindspeed
 + meanhumidity, data = dtrain,
    init.theta = 25.20504128, link = log)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-5.4755  -0.5420   0.0162   0.4864   3.5208

Coefficients:
                      Estimate Std. Error z value Pr(>|z|)
(Intercept)          7.6174644  0.0840575  90.622  < 2e-16 ***
Index                0.0022303  0.0001012  22.032  < 2e-16 ***
as.factor(season)2   0.2100812  0.0427576   4.913 8.96e-07 ***
as.factor(season)3   0.0052007  0.0540746   0.096  0.92338
as.factor(season)4   0.1061301  0.0389096   2.728  0.00638 **
as.factor(holiday)1 -0.0311585  0.0680255  -0.458  0.64692
meanatemp            0.0323311  0.0025503  12.677  < 2e-16 ***
meanwindspeed       -0.0114980  0.0024446  -4.703 2.56e-06 ***
meanhumidity        -0.0075700  0.0008730  -8.671  < 2e-16 ***
---
Signif. codes:  0    ***    0.001    **    0.01    *    0.05    .    0.1

(Dispersion parameter for Negative Binomial(25.205) family taken to be 1)

    Null deviance: 1666.47  on 305  degrees of freedom
Residual deviance:  308.95  on 297  degrees of freedom
```

```
AIC: 5007.3

Number of Fisher Scoring iterations: 1


              Theta:  25.21
          Std. Err.:  2.04

 2 x log-likelihood:  -4987.298
> p6<-predict(m6, newdata=dtest, type="response")
>
> RMSPE(dtest$Total, p6);
[1] 1013.551
```

The Gaussian distribution provides us with the least RMSPE and hence for subsequent analysis we continue with a Gaussian error distribution.

4.6 Variable Selection: Stepwise Regression Method

For building predictive models, selection of the right variables is important. The selection of the right variables can be achieved by many different methods. In this section we would learn about a stepwise regression fit for selecting the right variables. In the next section, we would look at a penalized regression method for selection of right variables.

In stepwise regression, the variables are included or excluded from the model in a stepwise manner depending on some information criterion. In stepwise regression usually Akaike's Information Criterion (AIC) or Bayesian Information Criterion (BIC) is used to select the right variables at each step. The information criterion measure the quality of model fit and can be used to compare two models. For example AIC measures the relative amount of information lost when a particular model is used to represent a data. Thus a lower AIC value is associated with a better model for the given data and variables.

Stepwise regression models can be estimated by using both a forward estimation method as well as a backward estimation method. In the forward estimation method, the estimation starts with the intercept only model. Then each of the variables are included in model individually and the AIC reduction by inclusion of each of the variables are measured. The variable which reduces the AIC the most is included in the next step. This process is continued till no further reduction in AIC is achieved. The set of variables included in the final step is the set of important variables. The backward selection method works in the exact reverse way. The estimation starts with the largest model where all variables are included. Then variables are excluded from the model in a stepwise manner similar to the forward selection method. The steps of the variable selection process based on AIC values can be seen in the stepwise estimation in R. The

command used for stepwise regression is *step*.

```
> m1<-glm(Total~1, data=dtrain, family="gaussian")
> m2<-glm(Total~Index+year+as.factor(month)+as.factor(day)
+as.factor(season)+as.factor(holiday)+as.factor(workingday)
+meanatemp+maxatemp+minatemp+sdatemp+meanhumidity
+maxhumidity+minhumidity+sdhumidity+meanwindspeed
+maxwindspeed+minwindspeed+sdwindspeed,
data=dtrain, family="gaussian")
>
> s1<-step(m1, scope=list(lower=m1, upper=m2),
+direction="forward")
Start:  AIC=5483.4
Total ~ 1

                        Df   Deviance     AIC
+ Index                  1  489119918  5245.5
+ year                   1  642006142  5328.7
+ as.factor(month)      11  619596596  5337.8
+ meanatemp              1  669065475  5341.3
+ maxatemp               1  690112563  5350.8
+ minatemp               1  693056654  5352.1
+ as.factor(season)      3  723262270  5369.2
+ meanwindspeed          1 1005582973  5466.0
+ sdhumidity             1 1017606899  5469.6
+ minhumidity            1 1017798714  5469.7
+ maxwindspeed           1 1027260881  5472.5
+ sdwindspeed            1 1045444205  5477.9
+ sdatemp                1 1046265871  5478.1
+ minwindspeed           1 1049146216  5479.0
+ meanhumidity           1 1052989477  5480.1
<none>                     1071405162  5483.4
+ maxhumidity            1 1068442734  5484.6
+ as.factor(holiday)     1 1070164355  5485.0
+ as.factor(workingday)  1 1071395753  5485.4
+ as.factor(day)        18 1041900916  5510.9

Step:  AIC=5245.46
Total ~ Index

                        Df   Deviance     AIC
+ as.factor(month)      11  196517044  4988.4
+ meanatemp              1  238343986  5027.5
+ maxatemp               1  242433884  5032.7
+ minatemp               1  262988630  5057.6
```

```
+ as.factor(season)       3 279004867 5079.7
+ sdhumidity              1 433748331 5210.7
+ minhumidity             1 446220524 5219.4
+ sdatemp                 1 455608868 5225.7
+ meanwindspeed           1 461136362 5229.4
+ minwindspeed            1 466911346 5233.2
+ meanhumidity            1 474545212 5238.2
+ maxwindspeed            1 477543820 5240.1
<none>                      489119918 5245.5
+ sdwindspeed             1 486944339 5246.1
+ as.factor(holiday)      1 487373865 5246.4
+ as.factor(workingday)   1 488451106 5247.0
+ year                    1 488808478 5247.3
+ maxhumidity             1 488964028 5247.4
+ as.factor(day)         18 477057117 5273.8

Step:  AIC=4988.43
Total ~ Index + as.factor(month)

                         Df  Deviance     AIC
+ minhumidity             1 154008770 4915.8
+ meanhumidity            1 165251461 4937.4
+ sdatemp                 1 177253594 4958.9
+ sdhumidity              1 178328656 4960.7
+ maxatemp                1 180820121 4965.0
+ meanatemp               1 182373993 4967.6
+ maxhumidity             1 188854974 4978.3
+ meanwindspeed           1 190334859 4980.6
+ maxwindspeed            1 190383250 4980.7
+ minwindspeed            1 192526769 4984.2
+ minatemp                1 192707201 4984.4
+ year                    1 194994410 4988.0
<none>                     196517044 4988.4
+ sdwindspeed             1 195350737 4988.6
+ as.factor(holiday)      1 195480154 4988.8
+ as.factor(workingday)   1 196078041 4989.7
+ as.factor(day)         18 187108512 5009.4

Step:  AIC=4915.84
Total ~ Index + as.factor(month) + minhumidity

                         Df  Deviance     AIC
+ meanatemp               1 134173619 4875.7
+ maxatemp                1 139786305 4888.2
+ minatemp                1 139930674 4888.5
+ maxwindspeed            1 140300332 4889.3
```

```
+ meanwindspeed            1 140361227 4889.5
+ sdwindspeed              1 148096036 4905.9
+ sdatemp                  1 148871237 4907.5
+ minwindspeed             1 149727111 4909.2
+ maxhumidity              1 151972792 4913.8
+ as.factor(holiday)       1 152158249 4914.1
+ sdhumidity               1 152544486 4914.9
+ year                     1 152783980 4915.4
<none>                       154008770 4915.8
+ as.factor(workingday)    1 153487264 4916.8
+ meanhumidity             1 153814322 4917.5
+ as.factor(day)          18 145848826 4935.2

Step:  AIC=4875.65
Total ~ Index + as.factor(month) + minhumidity
 + meanatemp

                         Df  Deviance     AIC
+ maxwindspeed            1 121347825 4846.9
+ meanwindspeed           1 122057247 4848.7
+ sdwindspeed             1 129280240 4866.3
+ minwindspeed            1 129524042 4866.9
+ maxatemp                 1 133209554 4875.4
<none>                       134173619 4875.7
+ as.factor(holiday)       1 133474535 4876.1
+ year                     1 133664418 4876.5
+ as.factor(workingday)    1 133783317 4876.8
+ sdatemp                  1 133854021 4876.9
+ meanhumidity             1 133863045 4876.9
+ minatemp                  1 133864589 4876.9
+ maxhumidity               1 133870586 4877.0
+ sdhumidity                1 134039238 4877.3
+ as.factor(day)           18 127043993 4894.9

Step:  AIC=4846.91
Total ~ Index + as.factor(month) + minhumidity
 + meanatemp + maxwindspeed

                         Df  Deviance     AIC
+ meanwindspeed           1 120197513 4846.0
+ minatemp                 1 120276048 4846.2
<none>                       121347825 4846.9
+ meanhumidity             1 120561396 4846.9
+ sdatemp                  1 120635315 4847.1
+ year                     1 120874428 4847.7
+ minwindspeed             1 120913716 4847.8
```

```
+ as.factor(holiday)      1 120920736 4847.8
+ sdwindspeed             1 121024137 4848.1
+ maxatemp                1 121027203 4848.1
+ as.factor(workingday)   1 121039351 4848.1
+ maxhumidity             1 121264995 4848.7
+ sdhumidity              1 121328455 4848.9
+ as.factor(day)         18 114076558 4864.0

Step:  AIC=4846
Total ~ Index + as.factor(month) + minhumidity
 + meanatemp +  maxwindspeed
 + meanwindspeed

                         Df  Deviance     AIC
+ meanhumidity            1 118710942 4844.2
+ minatemp                1 119173384 4845.4
<none>                     120197513 4846.0
+ sdatemp                 1 119531397 4846.3
+ year                    1 119620720 4846.5
+ as.factor(holiday)      1 119837603 4847.1
+ maxatemp                1 119857781 4847.1
+ as.factor(workingday)   1 119881813 4847.2
+ sdhumidity              1 120186977 4848.0
+ minwindspeed            1 120191678 4848.0
+ maxhumidity             1 120192783 4848.0
+ sdwindspeed             1 120194351 4848.0
+ as.factor(day)         18 113038539 4863.2

Step:  AIC=4844.19
Total ~ Index + as.factor(month) + minhumidity
 + meanatemp +  maxwindspeed
+ meanwindspeed + meanhumidity

                         Df  Deviance     AIC
+ maxhumidity             1 115761880 4838.5
+ sdhumidity              1 116724692 4841.0
+ minatemp                1 117399067 4842.8
+ sdatemp                 1 117561458 4843.2
<none>                     118710942 4844.2
+ year                    1 118220541 4844.9
+ as.factor(holiday)      1 118370147 4845.3
+ as.factor(workingday)   1 118394812 4845.4
+ maxatemp                1 118663656 4846.1
+ sdwindspeed             1 118696383 4846.1
+ minwindspeed            1 118701783 4846.2
+ as.factor(day)         18 111858168 4862.0
```

```
Step:  AIC=4838.49
Total ~ Index + as.factor(month) + minhumidity
 + meanatemp +  maxwindspeed
+ meanwindspeed + meanhumidity
+ maxhumidity

                          Df  Deviance    AIC
+ sdatemp                  1 114554195 4837.3
+ minatemp                 1 114813443 4838.0
<none>                       115761880 4838.5
+ year                     1 115304335 4839.3
+ as.factor(workingday)    1 115457042 4839.7
+ as.factor(holiday)       1 115589784 4840.0
+ minwindspeed             1 115731884 4840.4
+ sdhumidity               1 115758053 4840.5
+ sdwindspeed              1 115761609 4840.5
+ maxatemp                 1 115761781 4840.5
+ as.factor(day)          18 108744204 4855.4

Step:  AIC=4837.28
Total ~ Index + as.factor(month) + minhumidity
 + meanatemp + maxwindspeed
+ meanwindspeed + meanhumidity
+ maxhumidity + sdatemp

                          Df  Deviance    AIC
+ maxatemp                 1 113219994 4835.7
<none>                       114554195 4837.3
+ year                     1 114026477 4837.9
+ as.factor(workingday)    1 114335085 4838.7
+ as.factor(holiday)       1 114387239 4838.8
+ minwindspeed             1 114519753 4839.2
+ minatemp                 1 114538211 4839.2
+ sdwindspeed              1 114553264 4839.3
+ sdhumidity               1 114553427 4839.3
+ as.factor(day)          18 106930810 4852.2

Step:  AIC=4835.7
Total ~ Index + as.factor(month) + minhumidity
+ meanatemp + maxwindspeed
+ meanwindspeed + meanhumidity
+ maxhumidity + sdatemp + maxatemp

                          Df  Deviance    AIC
<none>                       113219994 4835.7
```

```
+ year                    1 112707986 4836.3
+ as.factor(workingday)   1 113005684 4837.1
+ as.factor(holiday)      1 113056127 4837.3
+ minatemp                1 113128934 4837.4
+ minwindspeed            1 113176407 4837.6
+ sdhumidity              1 113192457 4837.6
+ sdwindspeed             1 113217429 4837.7
+ as.factor(day)         18 105765293 4850.9
```

The summary of the final estimated model can be seen below.

```
> summary(s1)

Call:
glm(formula = Total ~ Index + as.factor(month) + minhumidity +
    meanatemp + maxwindspeed + meanwindspeed + meanhumidity +
    maxhumidity + sdatemp + maxatemp,
    family = "gaussian", data = dtrain)

Deviance Residuals:
     Min        1Q    Median        3Q       Max
-1941.79   -337.80     17.81    349.33   2741.56

Coefficients:
                   Estimate Std. Error t value Pr(>|t|)
(Intercept)      2029.1924   333.2830   6.088 3.67e-09 ***
Index               9.3216     0.3339  27.918  < 2e-16 ***
as.factor(month)2  -0.9885   195.9226  -0.005 0.995978
as.factor(month)3 464.5309   189.5604   2.451 0.014864 *
as.factor(month)4 730.1723   212.2721   3.440 0.000669 ***
as.factor(month)5 1630.1544  235.6765   6.917 3.04e-11 ***
as.factor(month)6 1259.8981  279.5295   4.507 9.60e-06 ***
as.factor(month)7 614.7673   318.1712   1.932 0.054327 .
as.factor(month)8 777.6623   303.8261   2.560 0.010997 *
as.factor(month)9 1127.0328  265.4441   4.246 2.95e-05 ***
as.factor(month)10 1051.7066 235.1751   4.472 1.12e-05 ***
as.factor(month)11 224.3401  195.6362   1.147 0.252459
as.factor(month)12 -95.4297  199.3113  -0.479 0.632450
minhumidity       -11.3989     7.6193  -1.496 0.135746
meanatemp         136.5020    38.3129   3.563 0.000430 ***
maxwindspeed      -14.0769     7.6856  -1.832 0.068057 .
meanwindspeed     -27.6100    13.1320  -2.102 0.036387 *
meanhumidity      -32.0009    11.1942  -2.859 0.004568 **
maxhumidity        14.9380     6.2593   2.387 0.017661 *
sdatemp           158.0982    62.5024   2.529 0.011963 *
maxatemp          -67.9193    37.0614  -1.833 0.067903 .
---
```

```
Signif. codes:  0    ***    0.001    **    0.01    *    0.05    .    0.1

(Dispersion parameter for gaussian family taken to be 397263.1)

    Null deviance: 1071405162  on 305  degrees of freedom
Residual deviance:  113219994  on 285  degrees of freedom
AIC: 4835.7

Number of Fisher Scoring iterations: 2

>
>
> p7<-predict(s1, newdata=dtest);
> plot(dtest$Index, dtest$Total,
+ pch="*", col=2, xlab="Day", ylab="Demand");
> points(dtest$Index, p7, pch="o", col=4);
> legend("topleft",
+ legend=c("Observed Demand", "Predicted Demand"),
+ pch=c("*","o"), col=c(2,4))
>
> RMSPE(dtest$Total, p7);
[1] 805.6834
```

We can observe that through variable selection, we have been able to reduce the prediction error. The RMSPE for the stepwise regression is 805.6834.

4.7 Least Absolute Shrinkage and Selection Operator (LASSO) for Variable Selection

Least Absolute Shrinkage and Selection Operator (LASSO) is a regression method for variable selection through penalization for high values of the regression coefficients. The LASSO is estimated by computing the following optimization.

$$\min \sum_{i=1}^{n} \{y_i - \beta_0 - \beta_1 x1_i - \cdots - \beta_k xk_i\}^2 \; s.t., \; \sum_{j=0}^{k} |\beta_j| < \lambda \qquad (11)$$

The least square error minimization is constrained by the sum of the absolute values of the regression parameters. For high values of the threshold parameter λ higher number of independent variables would be selected in the final model and vice-versa. The optimal value of lambda is computed such that the prediction error on a hold-out sample in the resulting model is minimized. The process of finding the optimal threshold value of λ by minimizing prediction error of hold-out sample is called cross-validation. LASSO is estimated by first estimating the optimal threshold parameter λ through cross-validation. Then, the optimal λ is used to estimate the final model. LASSO is estimated in R using a package

called **glmnet**. The function *cv.glmnet()* computes the optimal λ value. Also, the function *glmnet()* takes a vector of values for the dependent variable and a matrix of values for the independent variables. Hence, it is important to convert a R formula into a matrix using the command *model.matrix(formula)*. The estimation steps are shown below.

4.7.1 Estimation of Optimal Threshold Parameter *lambda* by Cross Validation

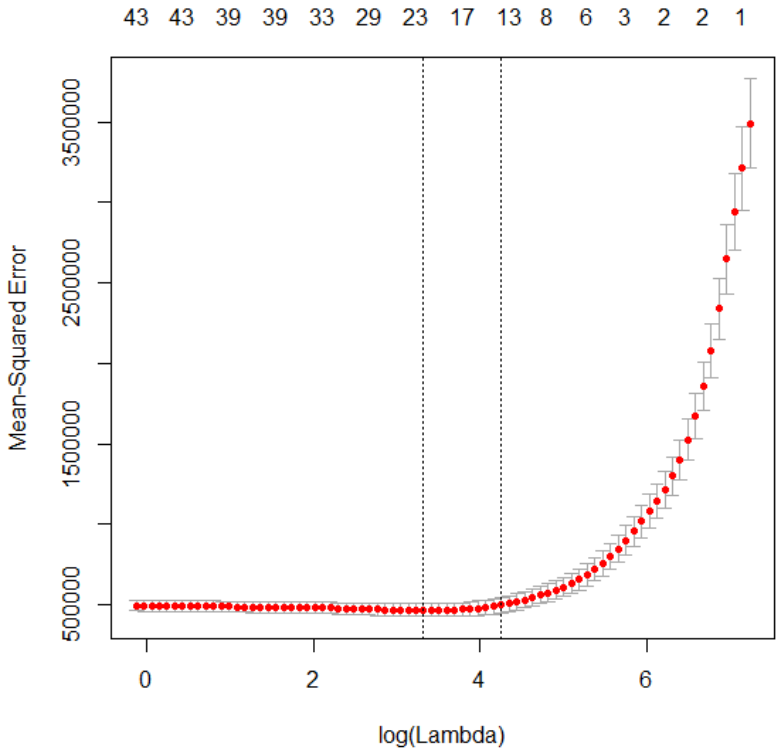
```
> library(glmnet)
> for1<-as.formula(Total~Index+year+as.factor(month)
+as.factor(day)+as.factor(season)+as.factor(holiday)
+as.factor(workingday)+meanatemp+maxatemp
+minatemp+sdatemp+meanhumidity+maxhumidity
+minhumidity+sdhumidity+meanwindspeed
+maxwindspeed+minwindspeed+sdwindspeed);
>
> y<-dtrain$Total;
> x<-model.matrix(for1, data=dtrain);
>
> l1<-cv.glmnet(x,y,alpha=1);
```

The value of lambda versus the mean squared prediction error can be plotted using the *plot* function (Figure 30. Also, the value of the optimal *lambda* can be printed as shown below.

```
> plot(l1)
>
> print(log(l1$lambda.min));
[1] 3.322026
```

So the optimal parameter value for penalizing the regression estimation for variable selection is 3.322026. This is computed for minimizing prediction error using cross-validation.

Figure 30: **Plot of the Lambda Values from the LASSO Cross-validation Estimation**



The final model is estimated using the optimal value of lambda as shown below. The non-zero values of beta parameters in the estimated model indicate the selected variables.

```
> l2<-glmnet(x,y, lambda=l1$lambda.min);
>
> l2$beta
49 x 1 sparse Matrix of class "dgCMatrix"
                                  s0
(Intercept)                        .
Index                          9.031173
year                               .
as.factor(month)2           -161.366241
as.factor(month)3             48.773160
as.factor(month)4                  .
as.factor(month)5            676.471227
```

```
as.factor(month)6          255.109152
as.factor(month)7                .
as.factor(month)8                .
as.factor(month)9          462.807145
as.factor(month)10         464.071254
as.factor(month)11               .
as.factor(month)12        -259.442281
as.factor(day)2                  .
as.factor(day)3                  .
as.factor(day)4             54.386960
as.factor(day)5                  .
as.factor(day)6                  .
as.factor(day)7                  .
as.factor(day)8           -208.284429
as.factor(day)9                  .
as.factor(day)10                 .
as.factor(day)11                 .
as.factor(day)12                 .
as.factor(day)13                 .
as.factor(day)14                 .
as.factor(day)15                 .
as.factor(day)16            39.930120
as.factor(day)17           214.056560
as.factor(day)18           -62.559451
as.factor(day)19                 .
as.factor(season)2         359.438586
as.factor(season)3               .
as.factor(season)4               .
as.factor(holiday)1              .
as.factor(workingday)1           .
meanatemp                   93.649176
maxatemp                         .
minatemp                         .
sdatemp                     26.141831
meanhumidity               -18.036155
maxhumidity                      .
minhumidity                -11.471582
sdhumidity                  15.739939
meanwindspeed              -26.705781
maxwindspeed               -10.749962
minwindspeed                     .
sdwindspeed                      .
> xn<-model.matrix(for1, data=dtest)
>
> p5<-predict(l2, newx=xn)
> RMSPE(dtest$Total, p5);
```

```
[1] 817.1028
>
```

As we can observe the LASSO variable selection method also achieves good prediction accuracy.

5 Machine Learning Methods for Predictive Modeling

Machine learning is a class of models that estimates functional relationship between the dependent variable and the independent variables without having to specify any parametric functional forms. The methods find out the functional relationships and functional forms from the data. In all previous models discussed so far, we have implicitly specified a functional form of the relationship between the dependent and the independent variables. In machine learning, data is the key and the model estimation starts with the data. Machine learning has the ability to uncover hidden patterns in a data. Due to the ability to recognize pattern from data, machine learning can be used for pattern recognition. Machine learning has been used to program self driving cars, face recognition in social media, on-line recommendations in Amazon and Netflix, and several other scientific and business application. Machine learning has been used in financial modeling. health-care applications, and supply chain automation and several other business functions to improve efficiency and effectiveness.

Several methods and algorithms of machine learning exist and are continuously being developed for improved power and performance. For the purpose of this document, we would be discussing two specific models, namely, **random forest** and **support vector machine (SVM)**.

5.1 Random Forest

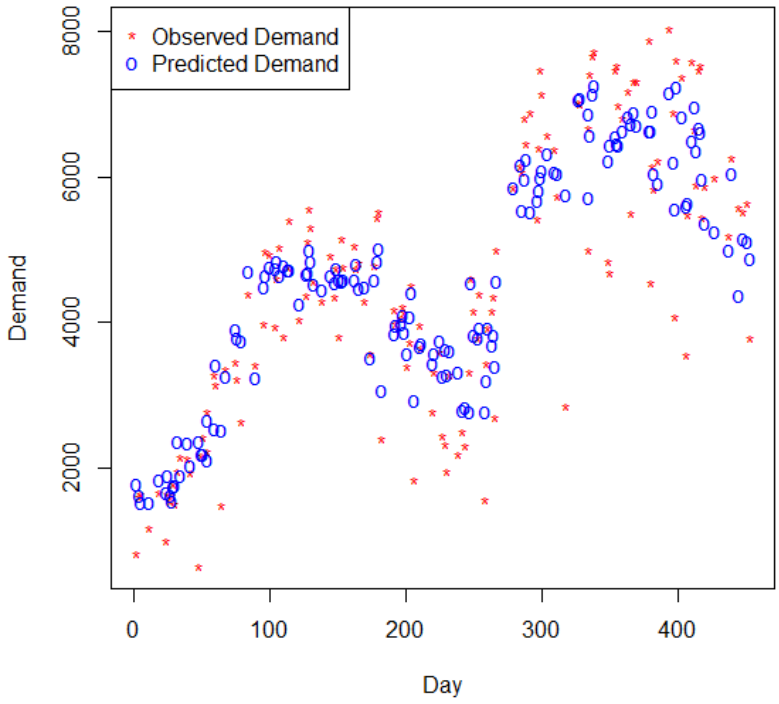
Random Forests are based on ensemble of decision trees. Decision trees are constructed by partitioning the data into hierarchical support spaces with a specific map of the dependent variable on each of the subspaces. Random forests are constructed by creating an ensemble of a number of random decision trees from the data by random partitioning of the data. The modal value of the dependent variable from the several decision trees in the ensemble is used as the best prediction of the dependent variable corresponding to a specific set of independent variable values. We refrain from theoretical discussion on random forests and decision trees for the purpose of this document since that would require some involved discussion. Instead, we will focus on constructing random forest models from the data in R.

Random Forests are estimated in R using the function *randomForest()* in the package **randomForest**. Code below shows construction of random forests in R.

```
> dtrain$month=as.factor(dtrain$month);
> dtrain$day=as.factor(dtrain$day);
> dtrain$season=as.factor(dtrain$season);
> dtrain$holiday=as.factor(dtrain$holiday);
> dtrain$workingday=as.factor(dtrain$workingday);
>
> r1<-randomForest(Total~Index+year+month+day+season
+holiday+workingday+meanatemp+maxatemp+minatemp
+sdatemp+meanhumidity+maxhumidity+minhumidity
+sdhumidity+meanwindspeed+maxwindspeed
+minwindspeed+sdwindspeed, data=dtrain, ntree=500,
+ do.trace=1, importance=TRUE, proximity=TRUE)
>
> dtest$month=as.factor(dtest$month);
> dtest$day=as.factor(dtest$day);
> dtest$season=as.factor(dtest$season);
> dtest$holiday=as.factor(dtest$holiday);
> dtest$workingday=as.factor(dtest$workingday);
>
> p6<-predict(r1, newdata=dtest, type="response");
>
> plot(dtest$Index, dtest$Total,
+ pch="*", col=2, xlab="Day", ylab="Demand");
> points(dtest$Index, p6, pch="o", col=4)
> legend("topleft",
+legend=c("Observed Demand", "Predicted Demand"),
+ pch=c("*","o"), col=c(2,4))
>
> RMSPE(dtest$Total, p6);
[1] 704.8452
```

As we can observe the use of machine learning (random forests) has helped in reducing the prediction error RMSPE substantially to 704.8452. The plot of the predicted values against the actual observed demand in the test set is shown for reference in Figure 31

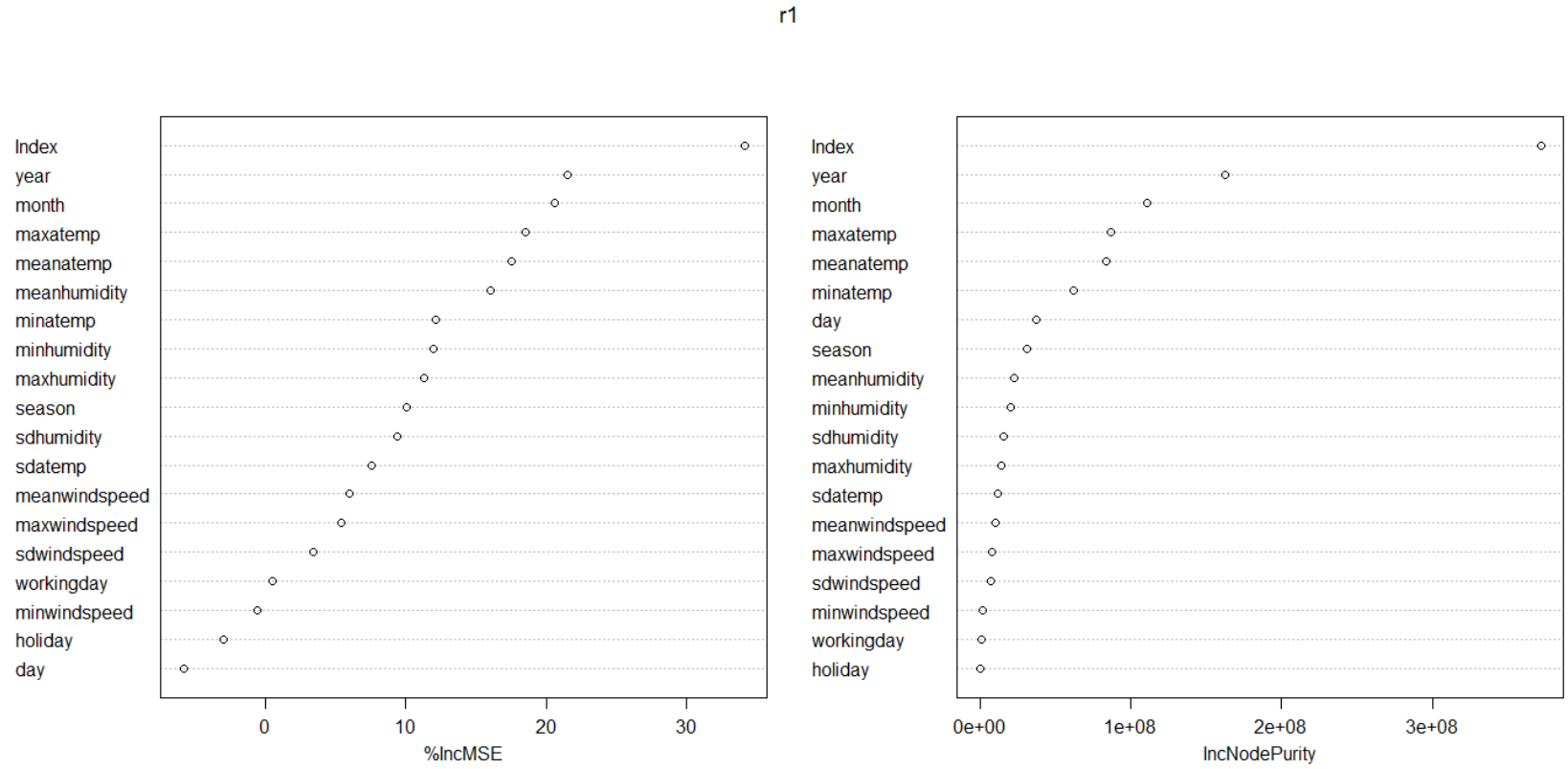
Figure 31: **Random Forest Prediction Model**



Random forest model can be used to infer variable importance by using the function *varImpPlot()* (Refer Figure 32.

```
> varImpPlot(r1)
```

Figure 32: **Random ForestVAriable Importance Plot**



It is clear that Index (trend) is the most predictive variable for the demand, followed by temperature, humidity, season, etc.

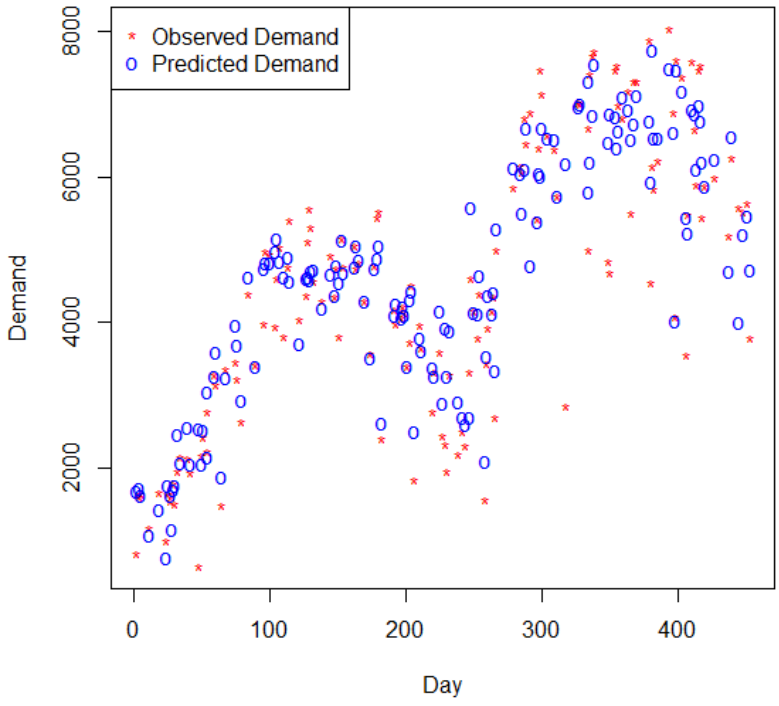
5.2 Support Vector Machines (SVM)

A support vector machine is a class of machine learning models which constructs separating hyperplanes (boundaries in high dimensions) to separate data pertaining to different classes or values of the dependent variable. Again, here we focus on the usage of SVM regression for constructing forecasting models. SVMs are constructed using the R function *svm()* in the library **e1071**.

```
> library(e1071)
>
> s1<-svm(for1, data=dtrain)
>
> p8<-predict(s1, newdata=dtest, type="response");
> plot(dtest$Index, dtest$Total, pch="*",
+ col=2, xlab="Day", ylab="Demand");
> points(dtest$Index, p8, pch="o", col=4)
> legend("topleft",
+ legend=c("Observed Demand", "Predicted Demand"),
+ pch=c("*","o"), col=c(2,4))
> RMSPE(dtest$Total, p8);
[1] 689.9342
```

As we can see the RMSPE has reduced further to 689.9342. The plot of the predicted values versus the actual observed demand shows close correspondence in the test set indicating that the predictive model fit has been fairly accurate (Refer Figure 33).

Figure 33: **Support Vector Machine (SVM) Prediction Model**



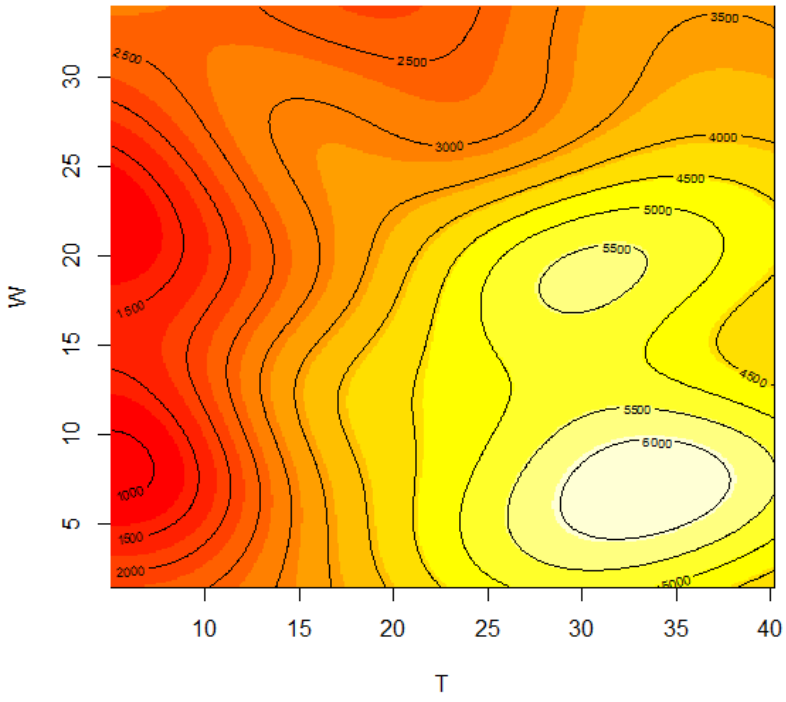
5.2.1 Understanding Machine :earning Models Through Interaction Plot of Two Variables

The working of the random forest and the SVM model can be understood by plotting the interaction of two independent variables, namely, mean temperature and mean wind speed to generate the demand surface contours as a function of these two dependent variables. Figure **??** shows the interaction contours of the demand forecasting model in SVM. The contour lines shows the mean demand on the iso-demand lines. For points that are not on the contours, a weighted average is computed based on the distance of the points from the mean contour lines.

```
> T<-seq(min(d$meanatemp), max(d$meanatemp), 0.1)
> W<-seq(min(d$meanwindspeed), max(d$meanwindspeed), 0.1)
>
> dn<-expand.grid(T,W);
> colnames(dn)<-c("meanatemp", "meanwindspeed");
```

```
>
> s2<-svm(Total~meanatemp+meanwindspeed, data=d);
>
> p9<-predict(s2, newdata=dn, type="response");
> z<-matrix(p9, nrow=length(T), ncol=length(W), byrow=FALSE);
>
> image(T,W,z)
> contour(T,W,z, add=TRUE)
```

Figure 34: **Support Vector Machine (SVM) Interaction Contour and Heat Map Plot**



6 Conclusion

In this chapter, we have studied different methods and models for demand forecasting in supply chains. Demand forecasting is a very important and complex function in any supply chains. We have discussed and learn time series based

methods, regression model based methods and finally machine learning (data mining) based methods for constructing demand forecasts. We have also seen how to compute forecast errors and use RMSPE forecast error measure to compare and select best forecasting model. In the context of regression models we have learn how to do variable selection and understand variable importance by different methods. Below, in Table 1 I summarize the methods and corresponding RMSPE values achieved for the demand forecasting problem of bike sharing program.

Table 1: Bike Demand Forecasting Methods

	Model	**RMSPE**
	Regression	
1	Linear Model: Intercept Only	1856.975
2	Linear Model: + Trend	1286.959
3	Linear Model: + Seasonality	1001.071
4	Linear Model: + Other Factors	864.287
5	GLM: Gaussian	864.287
6	GLM: Poisson	929.131
7	GLM: Neg. Binomial	1013.551
8	Stepwise Regression	805.683
9	LASSO Regression	817.103
	Machine Learning (Data Mining)	
10	Random Forest	704.845
11	SVM	689.934

7 Practice Exercise

Do the following separately for casual demand and registered demand separately:

1. Split the data into test set and train set.
2. Estimate the demand forecast for the test set for casual and registered users separately using: linear model, stepwise regression, random forest, and SVM.
3. Compute the RMSPE for each of the models. Which demand is easier to predict, causal or registered. Provide reasons for your observations.
4. Add the predicted demand for casual and registered demand to generate a prediction for total demand. Compute the RMSPE for total demand. This is a dis-aggregated method of demand forecasting.
5. Now generate the prediction for total demand using the data for total demand using the linear model, stepwise regression, random forest, and SVM. Compute the RMSPE. This is the aggregate demand forecast.
6. Which RMSPE for total demand is lower, aggregated method or disaggregated method. Provide justification for your observations.