Rental Bikes Analysis

Xinzhu Li

Importing data and data processing

```
setwd("/Users/lixinzhu/Desktop/RData/R-bike_rental")
bike_data <- read.csv("/Users/lixinzhu/Desktop/RData/R-bike_rental/hour.csv")

head(bike_data)
```

```
  instant     dteday season yr mnth hr holiday weekday workingday weathersit
1       1 2011-01-01      1  0    1  0       0       6          0          1
2       2 2011-01-01      1  0    1  1       0       6          0          1
3       3 2011-01-01      1  0    1  2       0       6          0          1
4       4 2011-01-01      1  0    1  3       0       6          0          1
5       5 2011-01-01      1  0    1  4       0       6          0          1
6       6 2011-01-01      1  0    1  5       0       6          0          2
  temp  atemp  hum windspeed casual registered cnt
1 0.24 0.2879 0.81    0.0000      3         13  16
2 0.22 0.2727 0.80    0.0000      8         32  40
3 0.22 0.2727 0.80    0.0000      5         27  32
4 0.24 0.2879 0.75    0.0000      3         10  13
5 0.24 0.2879 0.75    0.0000      0          1   1
6 0.24 0.2576 0.75    0.0896      0          1   1
```

```
str(bike_data)
```

```
'data.frame':   17379 obs. of  17 variables:
 $ instant    : int  1 2 3 4 5 6 7 8 9 10 ...
 $ dteday     : chr  "2011-01-01" "2011-01-01" "2011-01-01" "2011-01-01" ...
 $ season     : int  1 1 1 1 1 1 1 1 1 1 ...
 $ yr         : int  0 0 0 0 0 0 0 0 0 0 ...
```

```
 $ mnth      : int  1 1 1 1 1 1 1 1 1 1 ...
 $ hr        : int  0 1 2 3 4 5 6 7 8 9 ...
 $ holiday   : int  0 0 0 0 0 0 0 0 0 0 ...
 $ weekday   : int  6 6 6 6 6 6 6 6 6 6 ...
 $ workingday: int  0 0 0 0 0 0 0 0 0 0 ...
 $ weathersit: int  1 1 1 1 1 2 1 1 1 1 ...
 $ temp      : num  0.24 0.22 0.22 0.24 0.24 0.24 0.22 0.2 0.24 0.32 ...
 $ atemp     : num  0.288 0.273 0.273 0.288 0.288 ...
 $ hum       : num  0.81 0.8 0.8 0.75 0.75 0.75 0.8 0.86 0.75 0.76 ...
 $ windspeed : num  0 0 0 0 0 0.0896 0 0 0 0 ...
 $ casual    : int  3 8 5 3 0 0 2 1 1 8 ...
 $ registered: int  13 32 27 10 1 1 0 2 7 6 ...
 $ cnt       : int  16 40 32 13 1 1 2 3 8 14 ...
```

```r
summary(bike_data)
```

```
    instant         dteday              season            yr        
 Min.   :    1   Length:17379       Min.   :1.000   Min.   :0.0000  
 1st Qu.: 4346   Class :character   1st Qu.:2.000   1st Qu.:0.0000  
 Median : 8690   Mode  :character   Median :3.000   Median :1.0000  
 Mean   : 8690                      Mean   :2.502   Mean   :0.5026  
 3rd Qu.:13034                      3rd Qu.:3.000   3rd Qu.:1.0000  
 Max.   :17379                      Max.   :4.000   Max.   :1.0000  
      mnth             hr           holiday            weekday     
 Min.   : 1.000   Min.   : 0.00   Min.   :0.00000   Min.   :0.000  
 1st Qu.: 4.000   1st Qu.: 6.00   1st Qu.:0.00000   1st Qu.:1.000  
 Median : 7.000   Median :12.00   Median :0.00000   Median :3.000  
 Mean   : 6.538   Mean   :11.55   Mean   :0.02877   Mean   :3.004  
 3rd Qu.:10.000   3rd Qu.:18.00   3rd Qu.:0.00000   3rd Qu.:5.000  
 Max.   :12.000   Max.   :23.00   Max.   :1.00000   Max.   :6.000  
   workingday       weathersit         temp           atemp       
 Min.   :0.0000   Min.   :1.000   Min.   :0.020   Min.   :0.0000  
 1st Qu.:0.0000   1st Qu.:1.000   1st Qu.:0.340   1st Qu.:0.3333  
 Median :1.0000   Median :1.000   Median :0.500   Median :0.4848  
 Mean   :0.6827   Mean   :1.425   Mean   :0.497   Mean   :0.4758  
 3rd Qu.:1.0000   3rd Qu.:2.000   3rd Qu.:0.660   3rd Qu.:0.6212  
 Max.   :1.0000   Max.   :4.000   Max.   :1.000   Max.   :1.0000  
      hum           windspeed          casual         registered   
 Min.   :0.0000   Min.   :0.0000   Min.   :  0.00   Min.   :  0.0  
 1st Qu.:0.4800   1st Qu.:0.1045   1st Qu.:  4.00   1st Qu.: 34.0  
 Median :0.6300   Median :0.1940   Median : 17.00   Median :115.0  
 Mean   :0.6272   Mean   :0.1901   Mean   : 35.68   Mean   :153.8  
```

```
 3rd Qu.:0.7800   3rd Qu.:0.2537   3rd Qu.: 48.00   3rd Qu.:220.0
 Max.   :1.0000   Max.   :0.8507   Max.   :367.00   Max.   :886.0
      cnt
 Min.   :  1.0
 1st Qu.: 40.0
 Median :142.0
 Mean   :189.5
 3rd Qu.:281.0
 Max.   :977.0
```

```r
sum(is.na(bike_data))
```

```
[1] 0
```

```r
bike_data$dteday <- as.Date(bike_data$dteday)
```

Exploratory Data Analysis

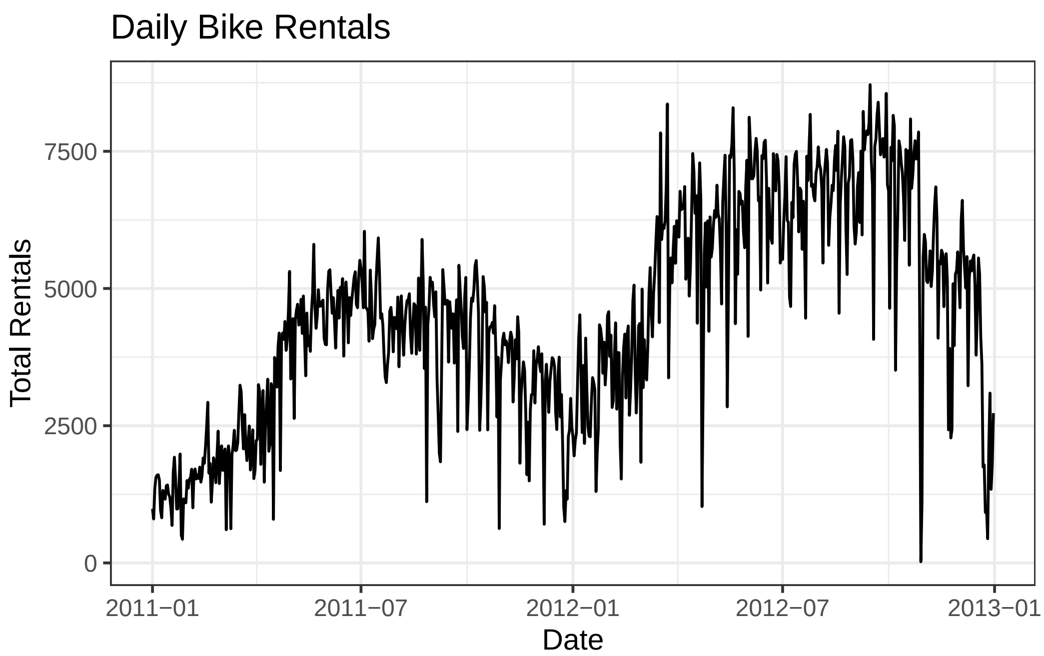
Numbers of rentals by day

By analyzing the daily frequency of bike rentals, it is evident that the time series exhibits clear trends and seasonality. Over the course of a year, the pattern shows an inverted U-shaped trend, with fewer rentals at the beginning and end of the year compared to the middle.

```r
daily_data <- aggregate(cnt ~ dteday, data = bike_data, sum)

library(dplyr)
```

```
Attaching package: 'dplyr'

The following objects are masked from 'package:stats':

    filter, lag

The following objects are masked from 'package:base':

    intersect, setdiff, setequal, union
```

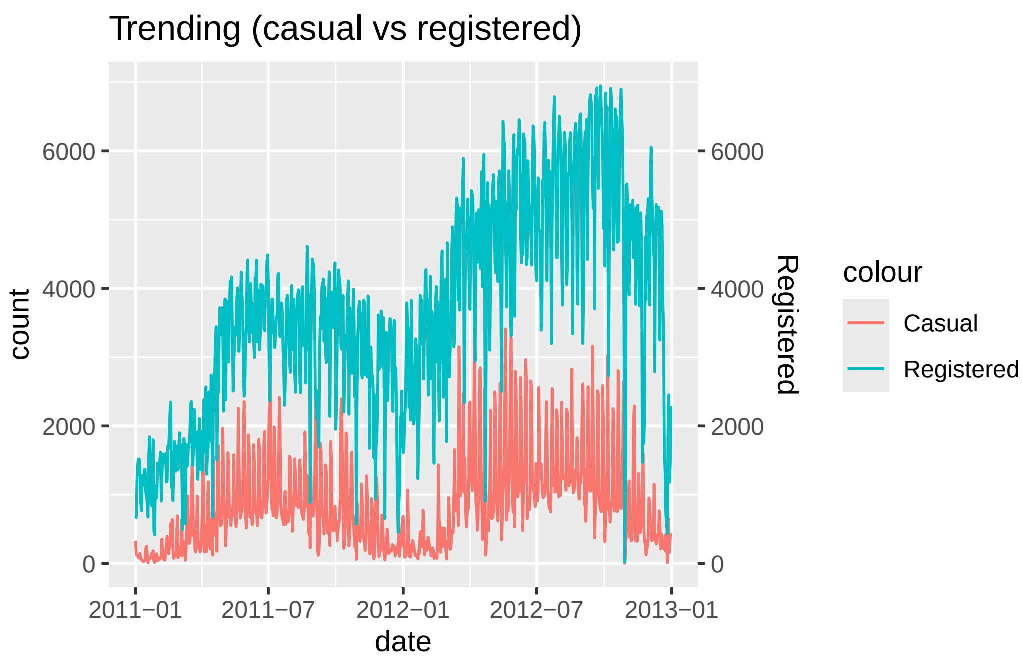
```r
library(ggplot2)
rental_cnt_by_day <- bike_data %>% group_by(dteday) %>% summarize(cnt_day= sum(cnt))

ggplot(rental_cnt_by_day, aes(x = dteday, y = cnt_day, group = 1)) +
  geom_line() +
  labs(title = "Daily Bike Rentals", x = "Date", y = "Total Rentals") +
  theme_bw()
```



Trends in casual and registered users

Breaking down the daily bike rental time series into casual users and registered users reveals that the majority of bike users are registered. Both registered and casual users also display an inverted U-shaped seasonal pattern.

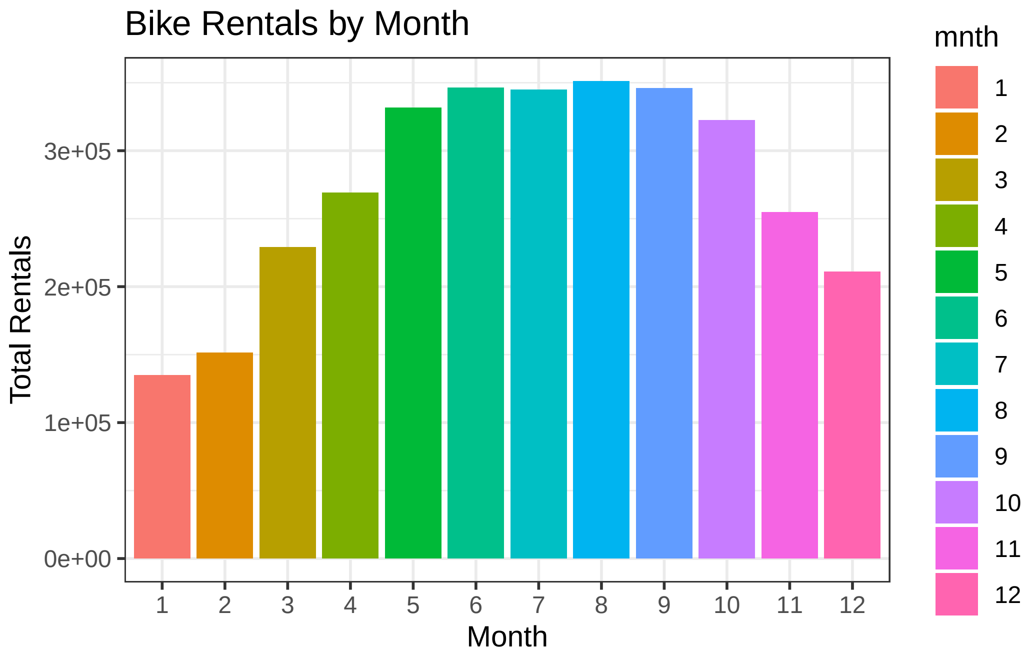
```r
library(dplyr)
daily_data <- bike_data %>% group_by(dteday) %>% summarise(casual_total= sum(casual),registe

ggplot(daily_data, aes(x = dteday)) +
  geom_line(aes(y = casual_total, color = "Casual")) +
  geom_line(aes(y = registered_total, color = "Registered")) +
  scale_y_continuous(sec.axis = sec_axis(~., name = "Registered")) +
  labs(title = "Trending (casual vs registered)", x = "date", y = "count")
```



Number of rentals in different months

Now let's analyze the monthly pattern. Across the 12 months of the year, we can clearly observe an inverted U-shaped trend as well. The peak in bike rental numbers occurs in May, June, July, August, and September, corresponding to the spring and summer seasons.

```
rental_cnt_by_month <- bike_data %>%
  group_by(mnth) %>%
  summarize(cnt_month = sum(cnt))

rental_cnt_by_month$mnth <- factor(rental_cnt_by_month$mnth, levels = 1:12)

ggplot(rental_cnt_by_month, aes(x = mnth, y = cnt_month, fill=mnth)) +
  geom_bar(stat = "identity") +
  labs(title = "Bike Rentals by Month", x = "Month", y = "Total Rentals") +
  scale_x_discrete(breaks = 1:12)+
  theme_bw()
```



Rental numbers by hour (including seasonal divisions)

Now let's analyze the hourly pattern. Over the 24 hours in a day, we can generally observe two distinct peaks: one in the morning and another in the afternoon. These peaks typically correspond to commuting times when people head to work and return home.

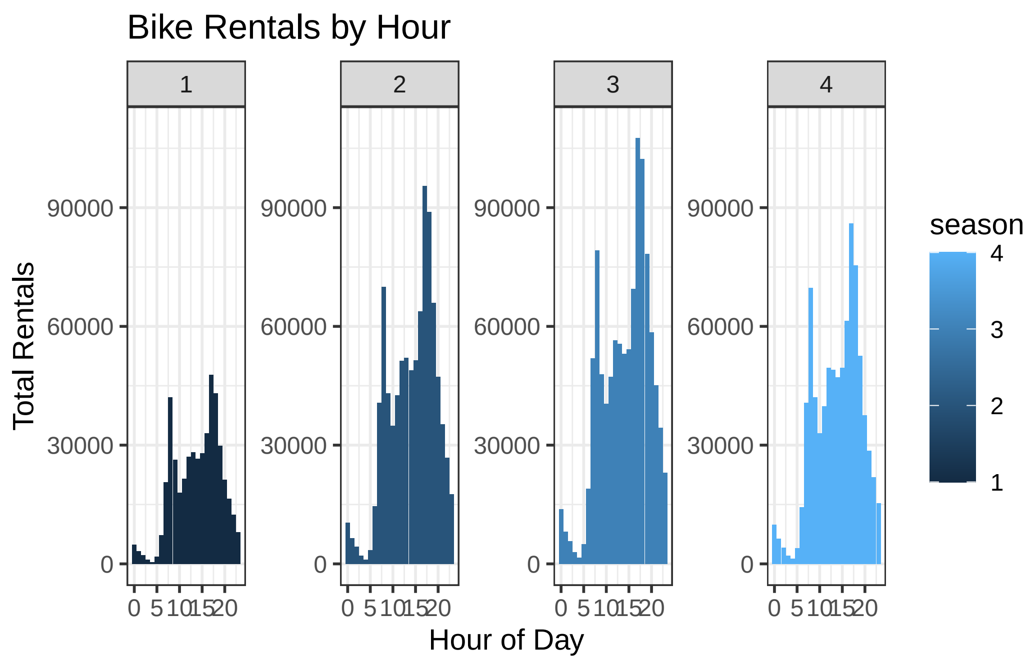
Additionally, we included a seasonal breakdown in our hourly analysis. Across the four seasons—spring, summer, fall, and winter—we found that the total number of bike rentals is generally higher in spring, summer, and fall compared to winter. Despite this variation in overall volume, the peak patterns remain consistent across seasons, aligning with morning and evening commuting times.

```r
rental_cnt_by_hour <- bike_data %>%
  group_by(hr, season) %>%
  summarize(cnt_hr = sum(cnt))
```

```
`summarise()` has grouped output by 'hr'. You can override using the `.groups`
argument.
```

```r
ggplot(rental_cnt_by_hour, aes(x = hr, y = cnt_hr,fill = season)) +
  geom_bar(stat = "identity") +
  facet_wrap(~ season, scales = "free", ncol = 4) +
```

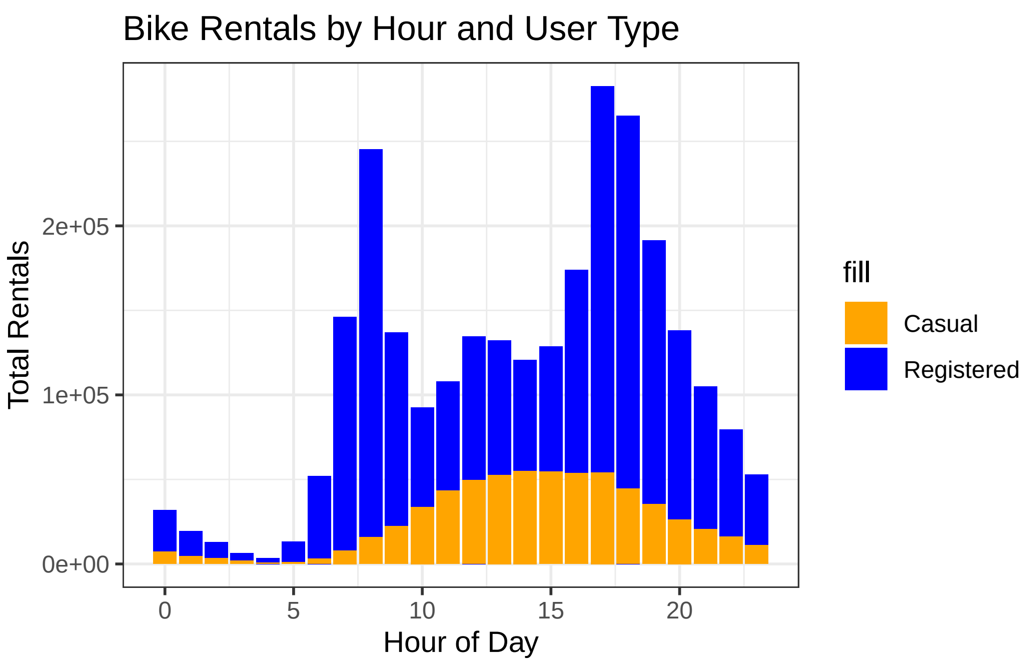
```
  labs(title = "Bike Rentals by Hour", x = "Hour of Day", y = "Total Rentals") +
  scale_y_continuous(limits = c(0, 110000)) +
  theme_bw()
```



The proportion of temporary users and registered users in different time periods

Now we are analyzing user data on an hourly basis, distinguishing between casual and registered users. The findings consistently show that, in general, registered users outnumber casual users across the entire day, regardless of the time period or specific hour.

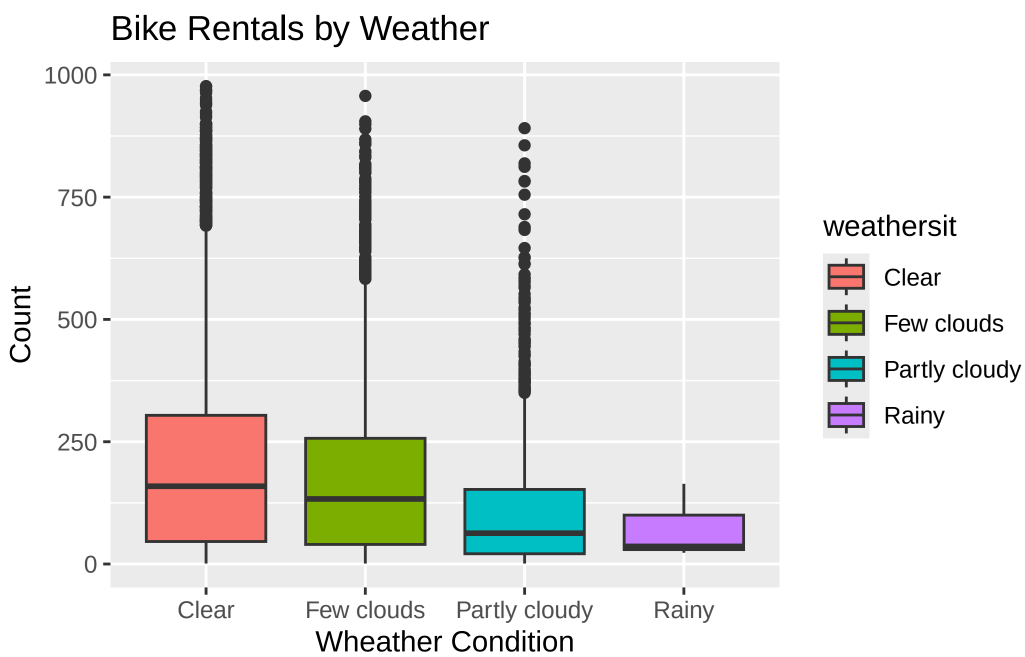
```
count_casual_registerd <- bike_data %>% group_by(hr) %>% summarise(cnt_registered = sum(regis

ggplot(count_casual_registerd, aes(x = hr)) +
  geom_bar(aes(y = cnt_registered, fill = "Registered"), stat = "identity", position = "stack
  geom_bar(aes(y = cnt_casual, fill = "Casual"), stat = "identity", position = "stack") +
  labs(title = "Bike Rentals by Hour and User Type", x = "Hour of Day", y = "Total Rentals")
  scale_fill_manual(values = c("Registered" = "blue", "Casual" = "orange")) +
  theme_bw()
```



Rental numbers in different weather conditions

Moving on to the weather conditions analysis, the data clearly indicates a higher number of users on clear and dry days, with a noticeable decrease in user activity during rainy days. This trend aligns with our expectations, as we would typically anticipate fewer users during adverse weather conditions such as rain.

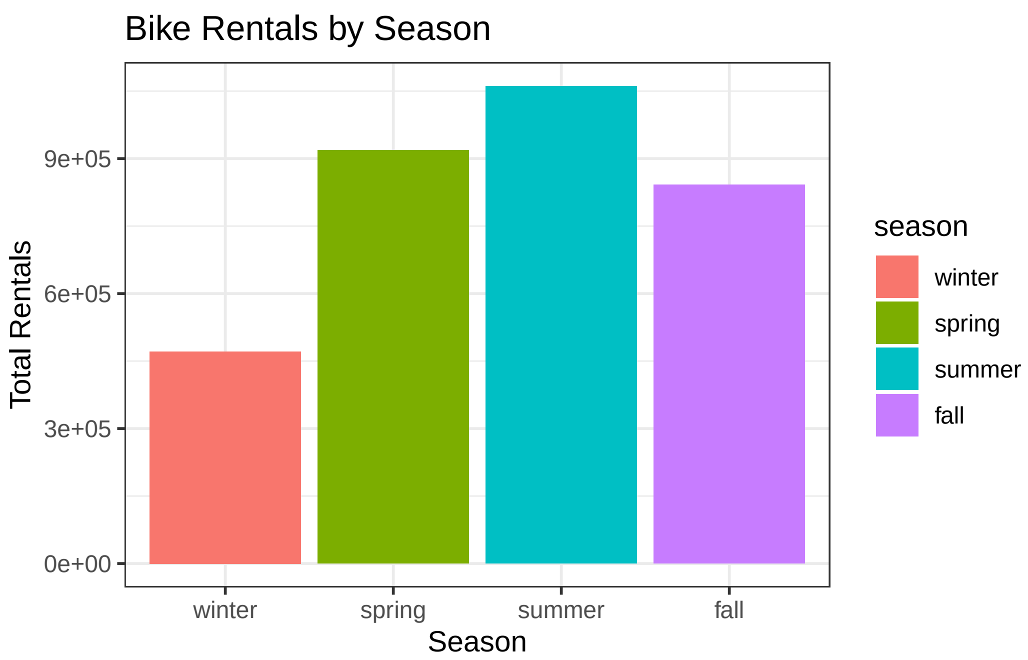
```
bike_data$weathersit <- as.factor(bike_data$weathersit)

bike_data$weathersit <- factor(bike_data$weathersit,
                               levels=c(1:4),
                               labels = c("Clear","Few clouds","Partly cloudy","Rainy"))

ggplot(bike_data,aes(x=weathersit, y=cnt,fill=weathersit))+
  geom_boxplot()+
  labs(x="Wheather Condition",
       y="Count",
       title="Bike Rentals by Weather")
```



Rental numbers in different seasons

Next, we turn to the seasonal analysis. In the previous hourly analysis, we observed that spring, summer, and fall generally have higher user activity compared to winter. This seasonal analysis further confirms that summer attracts the highest number of users and bike rentals, followed by spring, then fall, with winter having the lowest activity.

```
rental_cnt_by_season <- bike_data %>% group_by(season) %>% summarize(cnt_season= sum(cnt))

rental_cnt_by_season$season <- factor(rental_cnt_by_season$season,
                                      levels=c(1,2,3,4),
                                      labels =c("winter","spring","summer","fall"))
ggplot(rental_cnt_by_season,aes(x=season, y=cnt_season,fill = season))+
  geom_bar(stat = "identity")+
  labs(title = "Bike Rentals by Season", x = "Season", y = "Total Rentals")+
  theme_bw()
```



Seasons-ANOVA

The results indicate that at least one season has a significantly different average value compared to the others. Using the Honest Significant Difference (HSD) test, we can compare these differences between seasons and identify which ones are statistically significant.

- Spring, summer, and autumn all have significantly higher average values compared to winte
- Summer has a significantly higher average value than spring.
- Although autumn's average value is lower than spring, the difference is not statistically
- Finally, autumn has a significantly lower average value compared to summer.

This analysis clarifies the relationships and significant differences between the seasons.

```r
bike_data$season <- as.factor(bike_data$season)
aov_result <- aov(cnt ~ season, data = bike_data)
summary(aov_result)
```

```
                Df    Sum Sq  Mean Sq F value Pr(>F)    
season           3  37729358 12576453   409.2 <2e-16 ***
Residuals    17375 534032233    30736                   
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
TukeyHSD(aov_result )
```

```
  Tukey multiple comparisons of means
    95% family-wise confidence level

Fit: aov(formula = cnt ~ season, data = bike_data)

$season
          diff       lwr         upr     p adj
2-1  97.229500  87.54202 106.9169764 0.0000000
3-1 124.901668 115.26026 134.5430748 0.0000000
4-1  87.754288  77.96798  97.5405970 0.0000000
3-2  27.672168  18.12517  37.2191613 0.0000000
4-2  -9.475213 -19.16852   0.2180949 0.0581801
4-3 -37.147380 -46.79465 -27.5001142 0.0000000
```

Weather situations-ANOVA

The results show that at least one weather condition is significantly different from the average value of other seasons.

```r
aov_result2 <- aov(cnt ~ weathersit, data = bike_data)
summary(aov_result2)
```

```
                Df    Sum Sq Mean Sq F value Pr(>F)    
weathersit       3  12285030 4095010   127.2 <2e-16 ***
Residuals    17375 559476561   32200                   
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
TukeyHSD(aov_result2)
```

```
  Tukey multiple comparisons of means
    95% family-wise confidence level

Fit: aov(formula = cnt ~ weathersit, data = bike_data)

$weathersit
                                diff        lwr       upr     p adj
Few clouds-Clear           -29.70378  -37.79095 -21.61661 0.0000000
Partly cloudy-Clear        -93.28999 -106.26764 -80.31234 0.0000000
Rainy-Clear               -130.53594 -396.75339 135.68152 0.5886467
Partly cloudy-Few clouds  -63.58621  -77.60667 -49.56576 0.0000000
Rainy-Few clouds          -100.83216 -367.10249 165.43817 0.7648878
Rainy-Partly cloudy        -37.24595 -303.70965 229.21775 0.9841444
```

Time series analysis

S-ARIMA

An interesting point is that while the numbers generated by the model are crucial, the graphical validation provides a more intuitive understanding. From the predicted curve, we can observe that it aligns with the seasonal trends of the past, reflecting the upward half of a reverse U-shape. Moreover, the predicted trend is validated by the fact that the total rentals in the past two years have shown a year-on-year increase, indicating that the forecast for the third year should be slightly higher than that of the previous two years.

```r
daily_data <- aggregate(cnt ~ dteday, data = bike_data, sum)
head(daily_data)
```

```
      dteday  cnt
1 2011-01-01  985
2 2011-01-02  801
3 2011-01-03 1349
4 2011-01-04 1562
5 2011-01-05 1600
6 2011-01-06 1606
```

```r
library(xts)
```

```
Loading required package: zoo
```

```
Attaching package: 'zoo'

The following objects are masked from 'package:base':

    as.Date, as.Date.numeric


######################### Warning from 'xts' package ##########################
#                                                                              #
# The dplyr lag() function breaks how base R's lag() function is supposed to  #
# work, which breaks lag(my_xts). Calls to lag(my_xts) that you type or       #
# source() into this session won't work correctly.                            #
#                                                                              #
# Use stats::lag() to make sure you're not using dplyr::lag(), or you can add #
# conflictRules('dplyr', exclude = 'lag') to your .Rprofile to stop           #
# dplyr from breaking base R's lag() function.                                #
#                                                                              #
# Code in packages is not affected. It's protected by R's namespace mechanism #
# Set `options(xts.warn_dplyr_breaks_lag = FALSE)` to suppress this warning.  #
#                                                                              #
###############################################################################


Attaching package: 'xts'

The following objects are masked from 'package:dplyr':

    first, last
```

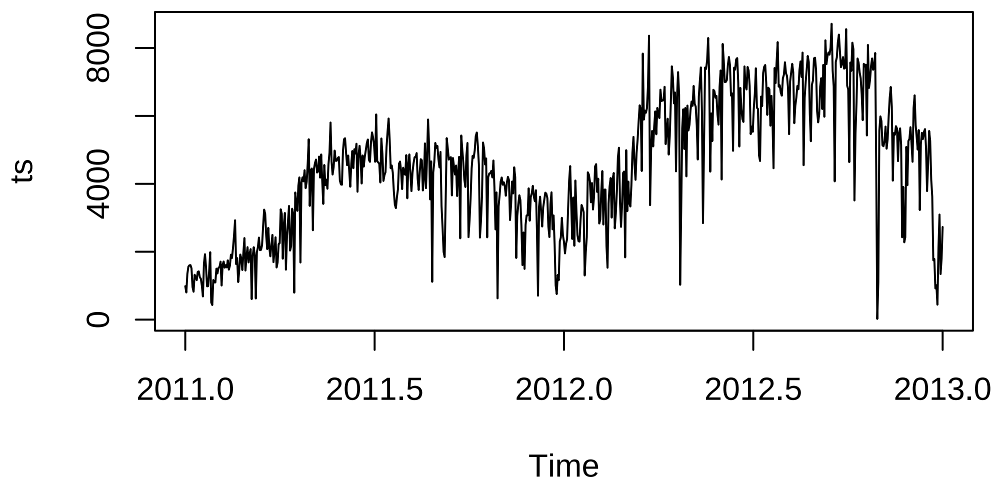
```r
library(tseries)
```

```
Registered S3 method overwritten by 'quantmod':
  method            from
  as.zoo.data.frame zoo
```

```r
str(daily_data)
```

```
'data.frame':   731 obs. of  2 variables:
 $ dteday: Date, format: "2011-01-01" "2011-01-02" ...
 $ cnt   : int  985 801 1349 1562 1600 1606 1510 959 822 1321 ...
```

```r
# Use ts function to create a time series
ts<- ts(daily_data$cnt, start = c(2011, 1), frequency = 365)
plot(ts)
```




```r
plot(ts)
adf.test(ts) # Non-stationary
```

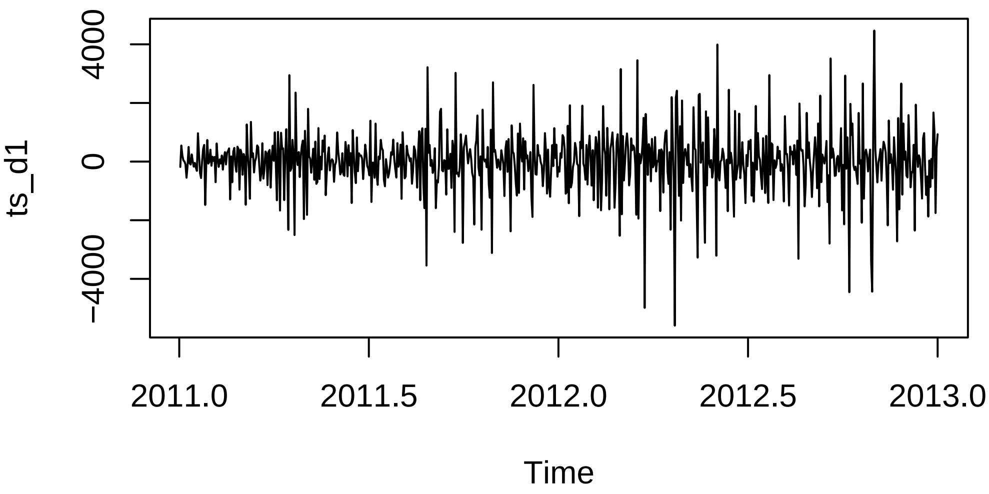
```
    Augmented Dickey-Fuller Test

data:  ts
Dickey-Fuller = -1.6351, Lag order = 9, p-value = 0.7327
alternative hypothesis: stationary
```

```r
ts_d1 <- diff(ts, differences = 1)
ts_d1 <- na.omit(ts_d1)
adf.test(ts_d1) # Stationary
```

```
Warning in adf.test(ts_d1): p-value smaller than printed p-value
```

```
    Augmented Dickey-Fuller Test

data:  ts_d1
Dickey-Fuller = -13.798, Lag order = 8, p-value = 0.01
alternative hypothesis: stationary
```

```
plot(ts_d1)
```

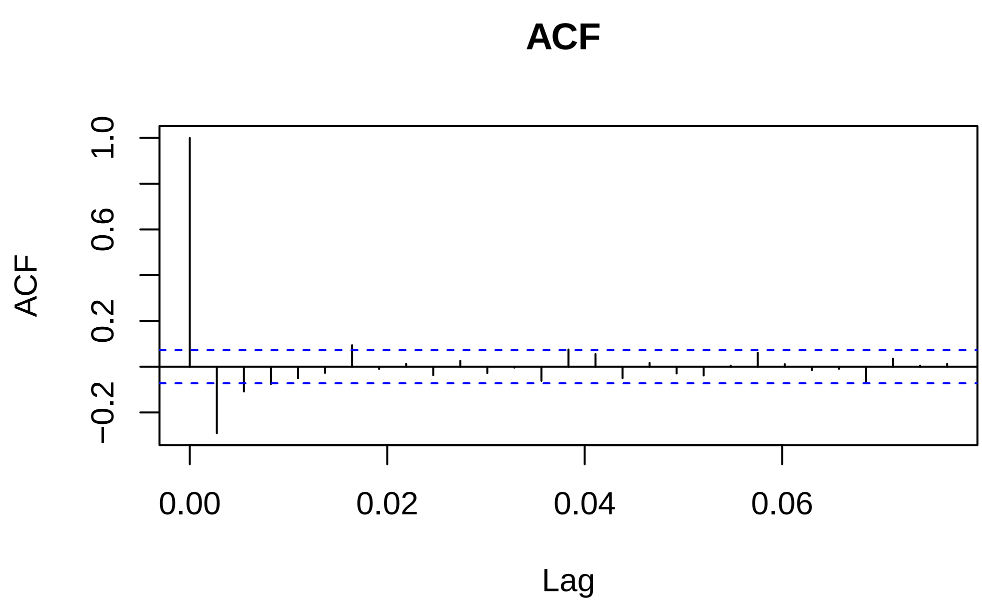



```
# White noise test - The result shows p < 0.05, reject the null hypothesis (the null hypothes
for(k in 1:4)print(Box.test(ts_d1,lag=1*k))
```

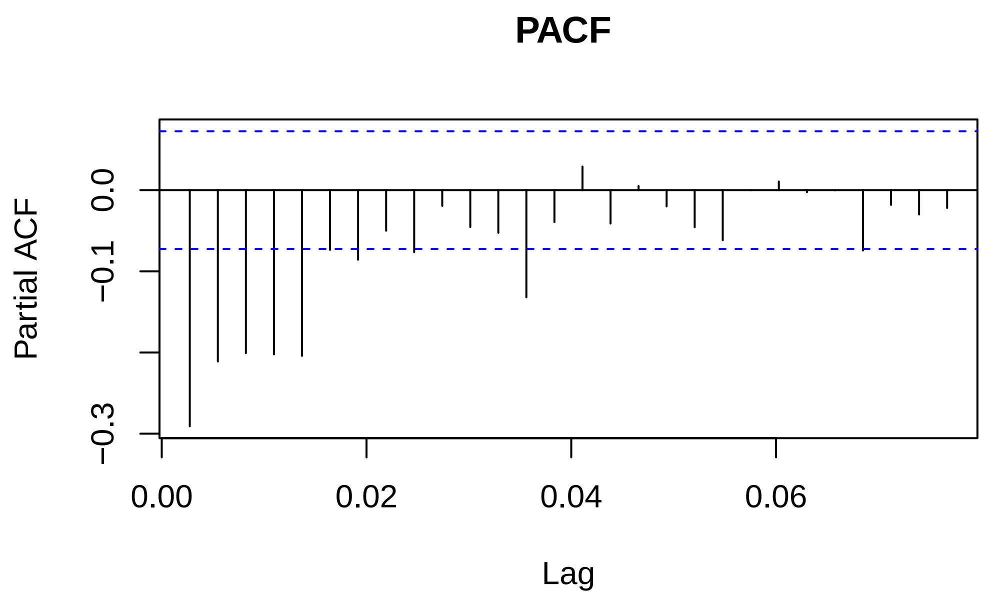
```
    Box-Pierce test

data:  ts_d1
X-squared = 61.804, df = 1, p-value = 3.775e-15


    Box-Pierce test

data:  ts_d1
```

```
X-squared = 70.405, df = 2, p-value = 5.551e-16


    Box-Pierce test

data:  ts_d1
X-squared = 74.62, df = 3, p-value = 4.441e-16


    Box-Pierce test

data:  ts_d1
X-squared = 76.5, df = 4, p-value = 9.992e-16
```

```
# Check ACF and PACF
acf(ts_d1, main = "ACF ")
```




```
pacf(ts_d1, main = "PACF ")
```



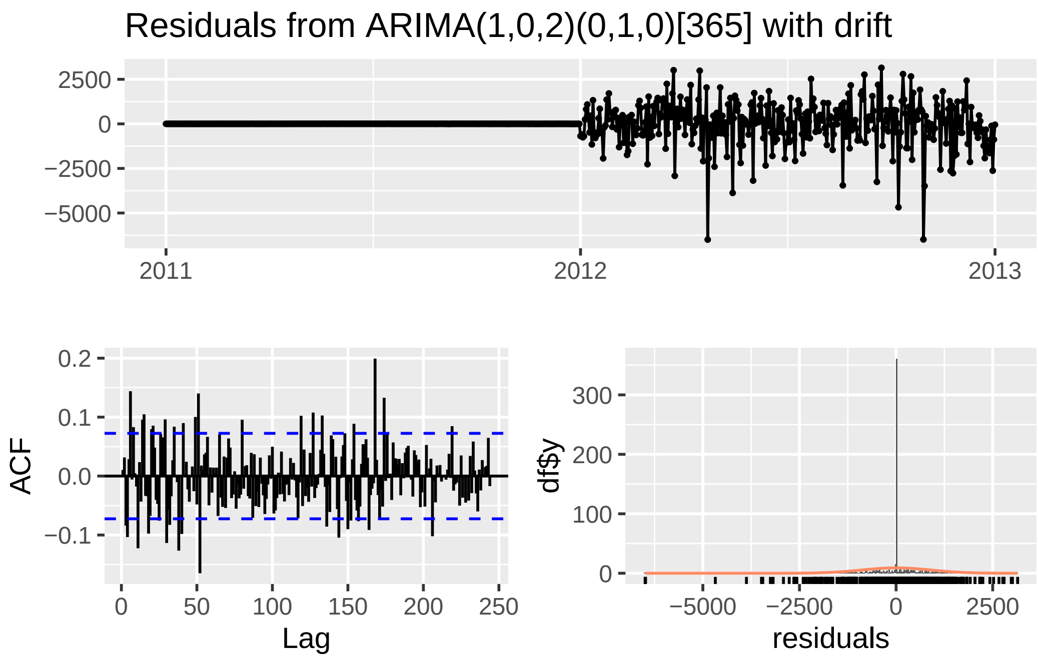

```
# ARIMA model fitting - Automatically selects the best model
library(forecast)
auto_model <- auto.arima(ts, seasonal = TRUE)
summary(auto_model)
```

```
Series: ts
ARIMA(1,0,2)(0,1,0)[365] with drift

Coefficients:
         ar1      ma1      ma2   drift
      0.9586  -0.6363  -0.1892  5.7093
s.e.  0.0283   0.0583   0.0506  0.7566

sigma^2 = 1599566:  log likelihood = -3131.76
AIC=6273.52   AICc=6273.68   BIC=6293.03

Training set error measures:
                   ME     RMSE      MAE       MPE     MAPE      MASE       ACF1
Training set 5.357076 890.0137 457.0405 -44.28372 51.73145 0.1967752 0.01047273
```

```
#Model diagnostics (white noise test, etc.) - The null hypothesis is white noise. The result
for(k in 1:3)print(Box.test(auto_model$residuals,lag=1*k))
```

```
    Box-Pierce test

data:  auto_model$residuals
X-squared = 0.080175, df = 1, p-value = 0.7771


    Box-Pierce test

data:  auto_model$residuals
X-squared = 0.81106, df = 2, p-value = 0.6666


    Box-Pierce test

data:  auto_model$residuals
X-squared = 5.9835, df = 3, p-value = 0.1124
```

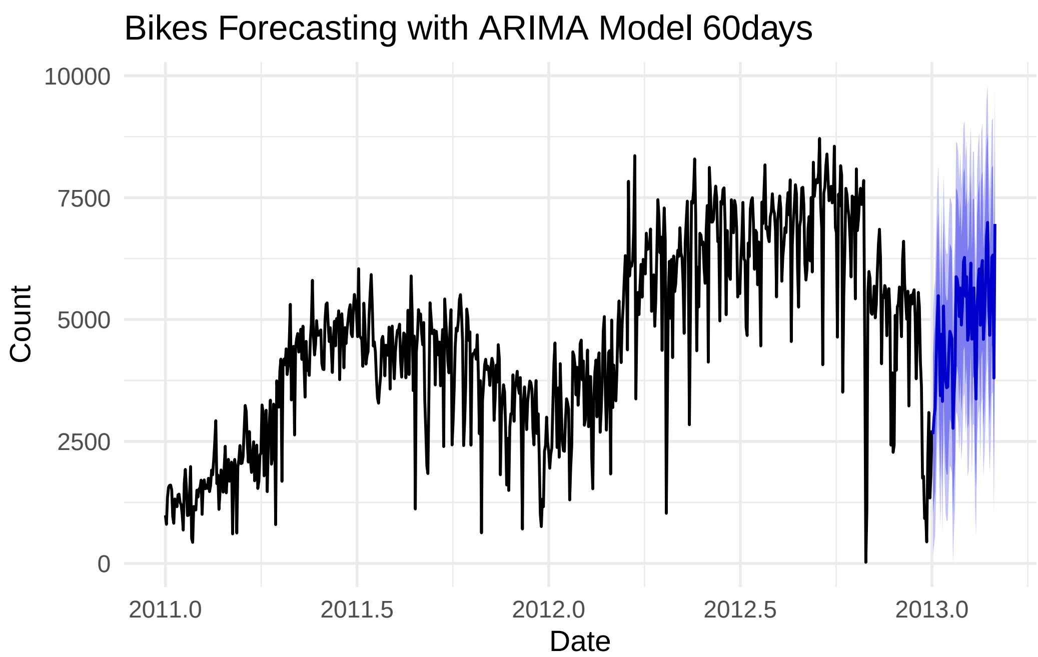
```r
checkresiduals(auto_model)
```




```
    Ljung-Box test
```

```
data:  Residuals from ARIMA(1,0,2)(0,1,0)[365] with drift
Q* = 377.27, df = 143, p-value < 2.2e-16

Model df: 3.   Total lags used: 146
```

```
#forecasting
forecast_values <- forecast(auto_model, h = 60)

autoplot(forecast_values) +
  xlab("Date") +
  ylab("Count") +
  ggtitle("Bikes Forecasting with ARIMA Model 60days") +
  theme_minimal()
```




```
show(forecast_values)
```

```
          Point Forecast      Lo 80    Hi 80       Lo 95    Hi 95
2013.0027       2649.939 1029.1105 4270.768  171.095544 5128.783
2013.0055       3001.034 1298.0890 4703.979  396.604253 5605.464
2013.0082       3187.601 1473.6264 4901.575  566.303332 5808.898
2013.0110       4143.909 2419.8622 5867.956 1507.206786 6780.612
```

2013.0137	5020.054	3286.8016	6753.306	2369.273350	7670.834
2013.0164	5491.124	3749.4553	7232.792	2827.471669	8154.776
2013.0192	4441.205	2691.8378	6190.572	1765.778813	7116.631
2013.0219	3436.379	1679.9673	5192.791	750.179020	6122.579
2013.0247	4700.726	2937.8649	6463.587	2004.662628	7396.790
2013.0274	3320.321	1551.5539	5089.088	615.225172	6025.417
2013.0301	5279.236	3505.0591	7053.413	2565.866602	7992.606
2013.0329	4433.541	2654.4073	6212.675	1712.590619	7154.492
2013.0356	3748.303	1964.6258	5531.980	1020.404101	6476.202
2013.0384	3600.585	1812.7430	5388.427	866.316609	6334.853
2013.0411	3620.449	1828.7879	5412.110	880.339959	6360.558
2013.0438	4288.953	2493.7897	6084.116	1543.487846	7034.418
2013.0466	4760.153	2961.7780	6558.529	2009.775529	7510.531
2013.0493	4705.104	2903.7820	6506.427	1950.219547	7459.989
2013.0521	4603.858	2799.8313	6407.884	1844.837474	7362.878
2013.0548	2768.463	961.9551	4574.970	5.647713	5531.278
2013.0575	3469.967	1661.1821	5278.752	703.669304	6236.265
2013.0603	3949.416	2138.5411	5760.291	1179.921770	6718.910
2013.0630	5879.854	4067.0599	7692.647	3107.424974	8652.282
2013.0658	5833.321	4018.7664	7647.876	3058.198999	8608.444
2013.0685	5659.860	3843.6875	7476.032	2882.264043	8437.455
2013.0712	5061.507	3243.8498	6879.164	2281.640326	7841.373
2013.0740	5648.300	3829.2792	7467.320	2866.348013	8430.251
2013.0767	4887.273	3067.0011	6707.546	2103.407146	7671.140
2013.0795	5286.462	3465.0402	7107.885	2500.837584	8072.087
2013.0822	6188.899	4366.4206	8011.377	3401.658958	8976.139
2013.0849	6275.614	4452.1657	8099.062	3486.890621	9064.337
2013.0877	5473.637	3649.2983	7297.976	2683.551619	8263.723
2013.0904	5878.998	4053.8405	7704.155	3087.660660	8670.335
2013.0932	4574.723	2748.8138	6400.631	1782.236085	7367.209
2013.0959	4703.838	2877.2391	6530.438	1910.295853	7497.381
2013.0986	5554.370	3727.1365	7381.604	2759.857523	8348.883
2013.1014	6158.342	4330.5257	7986.158	3362.938240	8953.745
2013.1041	4597.777	2769.4255	6426.128	1801.554726	7393.999
2013.1068	5637.698	3808.8544	7466.541	2840.723276	8434.672
2013.1096	5650.125	3820.8301	7479.420	2852.459750	8447.791
2013.1123	3999.080	2169.3696	5828.790	1200.779571	6797.380
2013.1151	3369.581	1539.4897	5199.673	570.697729	6168.465
2013.1178	5272.648	3442.2062	7103.090	2473.228776	8072.068
2013.1205	5782.299	3951.5347	7613.062	2982.386834	8582.210
2013.1233	6038.550	4207.4901	7869.609	3238.185593	8838.914
2013.1260	4883.418	3052.0867	6714.750	2082.638322	7684.198
2013.1288	6040.920	4209.3384	7872.501	3239.757871	8842.081

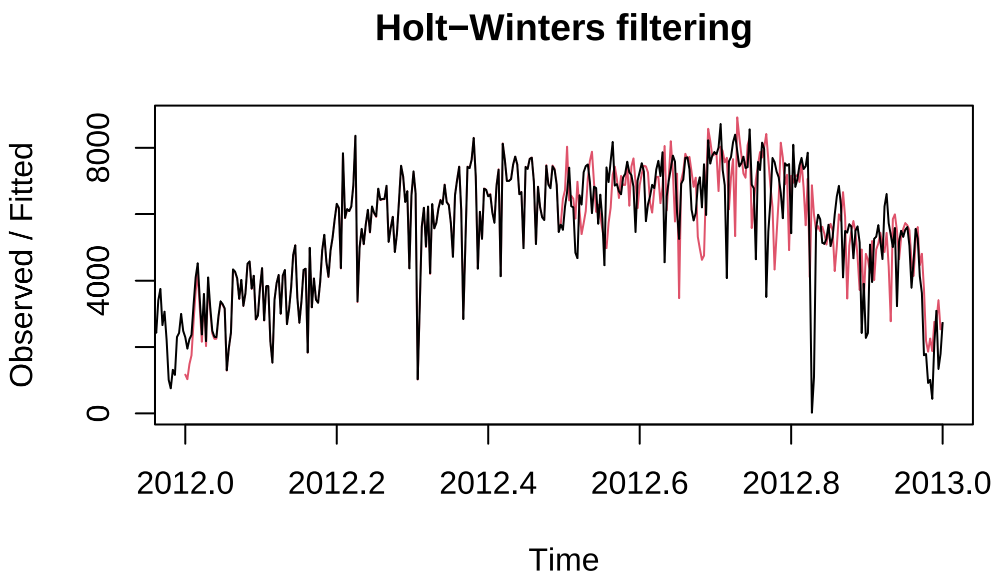
```
2013.1315       6213.069 4381.2587 8044.880 3411.556678 9014.582
2013.1342       4591.882 2759.8604 6423.904 1790.046777 7393.717
2013.1370       5039.371 3207.1560 6871.587 2237.239804 7841.503
2013.1397       5694.551 3862.1574 7526.944 2892.147009 8496.955
2013.1425       6697.433 4864.8763 8529.990 3894.779259 9500.087
2013.1452       6993.031 5160.3236 8825.738 4190.147049 9795.915
2013.1479       5424.356 3591.5102 7257.201 2621.260515 8227.451
2013.1507       4675.419 2842.4463 6508.391 1872.129435 7478.708
2013.1534       5338.231 3505.1419 7171.320 2534.763247 8141.698
2013.1562       6276.802 4443.6064 8109.999 3473.171051 9080.434
2013.1589       6323.144 4489.8491 8156.438 3519.361623 9126.926
2013.1616       3799.264 1965.8788 5632.649  995.343422 6603.184
2013.1644       6960.172 5126.7040 8793.640 4156.124601 9764.220
```

Holt-winters Three-Parameter Exponential Smoothing

The black line in the first chart represents the original input time series, while the red line illustrates the model's fit to the training dataset.

Based on the previous S-ARIMA model and the Holt-Winters three-parameter model, we observe that both models predict the same general upward trend for the third year. However, we might consider the S-ARIMA model as more suitable for this data, as it accounts for both seasonality, which follows a reversed U-shape, and a trend, with a year-on-year increase in bike rentals. Given this, the predicted data for the first part of the third year should logically be higher than that of the second year. From a quick visual comparison of the charts, it appears that the S-ARIMA model might be a better fit for the data.

```r
x.fit <- HoltWinters(ts)
plot(x.fit)
```

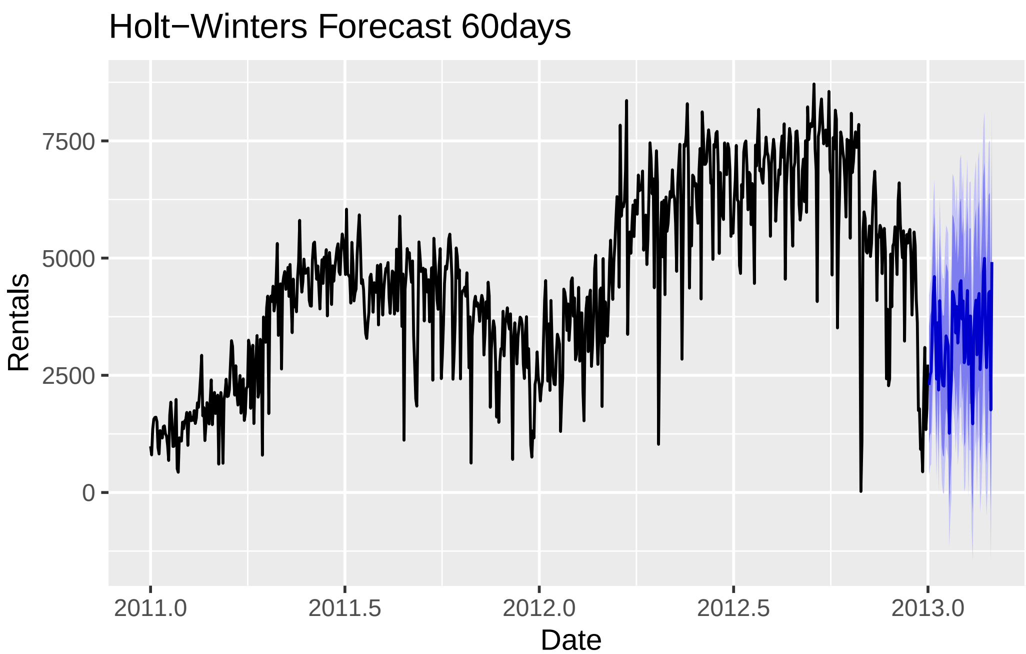



```r
x.fore <- forecast(x.fit,h=60)
x.fore
```

```
          Point Forecast      Lo 80    Hi 80        Lo 95    Hi 95
2013.0027       2294.580 1042.7153 3546.444   380.018305 4209.141
2013.0055       2506.322 1234.5346 3778.109   561.291135 4451.353
2013.0082       2575.884 1284.4815 3867.287   600.854225 4550.914
2013.0110       3430.095 2119.3701 4740.820  1425.514307 5434.675
2013.0137       4215.569 2885.8032 5545.335  2181.867574 6249.271
2013.0164       4606.858 3258.3191 5955.396  2544.446006 6669.269
2013.0192       3487.742 2120.6888 4854.795  1397.014637 5578.469
2013.0219       2417.752 1032.4316 3803.072   299.087348 4536.416
2013.0247       3622.071 2218.7209 5025.420  1475.832377 5768.309
2013.0274       2185.429  764.2783 3606.579    11.966648 4358.891
2013.0301       4092.110 2653.3787 5530.841  1891.760492 6292.459
2013.0329       3198.814 1742.7151 4654.914   971.902621 5425.726
2013.0356       2470.776  997.5130 3944.039   217.614784 4723.937
2013.0384       2285.396  795.1670 3775.625     6.287518 4564.504
2013.0411       2268.607  761.6035 3775.611   -36.156042 4573.370
2013.0438       2901.704 1378.1105 4425.298   571.568597 5231.840
2013.0466       3341.211 1801.2054 4881.216   985.975779 5696.446
2013.0493       3254.942 1698.6979 4811.185   874.872154 5635.011
2013.0521       3124.368 1552.0533 4696.682   719.720258 5529.015
```

2013.0548	1262.695	-325.5280	2850.917	-1166.282344	3691.672
2013.0575	1935.059	331.0855	3539.032	-518.006519	4388.124
2013.0603	2387.323	767.7530	4006.894	-89.595661	4864.242
2013.0630	4290.990	2655.9714	5926.009	1790.444808	6791.535
2013.0658	4218.804	2568.4817	5869.127	1694.853715	6742.755
2013.0685	4022.612	2357.1261	5688.097	1475.471274	6569.752
2013.0712	3402.235	1721.7228	5082.747	832.113429	5972.356
2013.0740	3967.531	2272.1257	5662.936	1374.632370	6560.430
2013.0767	3186.942	1476.7730	4897.111	571.464376	5802.419
2013.0795	3566.550	1841.7441	5291.356	928.686995	6204.413
2013.0822	4449.956	2710.6358	6189.276	1789.895444	7110.016
2013.0849	4518.659	2764.9451	6272.373	1836.585079	7200.733
2013.0877	3697.984	1929.9937	5465.975	994.076052	6401.892
2013.0904	4086.731	2304.5783	5868.884	1361.163623	6812.299
2013.0932	2767.517	971.3139	4563.721	20.461197	5514.573
2013.0959	2883.689	1073.5437	4693.834	115.310823	5652.067
2013.0986	3719.171	1895.1909	5543.151	929.634137	6508.708
2013.1014	4308.898	2471.1872	6146.609	1498.361802	7119.435
2013.1041	2734.571	883.2307	4585.911	-96.809503	5565.951
2013.1068	3761.814	1896.9446	5626.684	909.742396	6613.886
2013.1096	3764.343	1886.0414	5642.644	891.728681	6636.957
2013.1123	2103.659	212.0211	3995.297	-789.351640	4996.670
2013.1151	1463.716	-441.1648	3368.598	-1449.548082	4376.981
2013.1178	3355.740	1437.7066	5273.773	422.361161	6289.118
2013.1205	3855.489	1924.3940	5786.585	902.133916	6808.845
2013.1233	4101.964	2157.8945	6046.034	1128.766141	7075.163
2013.1260	2936.960	980.0019	4893.918	-55.949271	5929.870
2013.1288	4084.209	2114.4467	6053.971	1071.717416	7096.701
2013.1315	4245.726	2263.2423	6228.210	1213.778748	7277.674
2013.1342	2620.453	625.3290	4615.577	-430.825937	5671.732
2013.1370	3060.085	1052.4003	5067.770	-10.403868	6130.574
2013.1397	3708.226	1688.0585	5728.394	618.646431	6797.806
2013.1425	4705.517	2672.9432	6738.090	1596.963786	7814.070
2013.1452	4994.281	2949.3772	7039.186	1866.870253	8121.693
2013.1479	3416.857	1359.6955	5474.018	270.700210	6563.013
2013.1507	2662.945	593.5995	4732.290	-501.845730	5827.735
2013.1534	3317.228	1235.7701	5398.686	133.912644	6500.544
2013.1562	4249.413	2155.9119	6342.914	1047.679374	7451.146
2013.1589	4289.879	2184.4044	6395.354	1069.833253	7509.925
2013.1616	1760.273	-357.1081	3877.654	-1477.981938	4998.528
2013.1644	4915.454	2786.2328	7044.674	1659.091350	8171.816

```r
autoplot(x.fore ) +
  labs(title = "Holt-Winters Forecast 60days", x = "Date", y = "Rentals")
```



Time series model comparison

```r
accuracy(forecast(auto_model))
```

```
                   ME     RMSE      MAE       MPE     MAPE      MASE       ACF1
Training set 5.357076 890.0137 457.0405 -44.28372 51.73145 0.1967752 0.01047273
```

```r
accuracy(forecast(x.fit))
```

```
                    ME     RMSE     MAE       MPE     MAPE      MASE      ACF1
Training set -19.57782 975.6961 508.292 -89.23609 97.35388 0.2188411 0.2305566
```

```r
#SARIMA better
```