Project Reproducibility Course 2020

Ida Hansson

November 6, 2020

# Abstract

The task of this project was to describe the relationships between number of total bike rentals and variables that seem to be of importance. From the results of this analysis the hour of the day seems to be of importance for how many bikes that are being rent.

# Material and Methods

The data used in this analysis is called *bikes* and is automatically loaded with the package *bikerentals*. The data set consist of hourly rental data, of the first 19 days of each month, spanning two years.

#### Data Fields

* *datetime* - hourly date + timestamp
* *season* -  
  1 = spring, 2 = summer, 3 = fall, 4 = winter
* *holiday* - whether the day is considered a holiday
* *workingday* - whether the day is neither a weekend nor holiday
* *weather* -  
  1: Clear, Few clouds, Partly cloudy, Partly cloudy  
  2: Mist + Cloudy, Mist + Broken clouds, Mist + Few clouds, Mist  
  3: Light Snow, Light Rain + Thunderstorm + Scattered clouds, Light Rain + Scattered clouds  
  4: Heavy Rain + Ice Pallets + Thunderstorm + Mist, Snow + Fog
* *temp* - temperature in Celsius
* *atemp* - “feels like” temperature in Celsius
* *humidity* - relative humidity
* *windspeed* - wind speed
* *casual* - number of non-registered user rentals initiated
* *registered* - number of registered user rentals initiated
* *count* - number of total rentals

#### Install and load packages

First we need to install and load all the packages needed. Start install the packages *tidyverse*, *lubridate*, *randomForest* and *devtools*, if you don´t already have them installed. Then you need to load the *devtools* package before you can install the *bikerentals* package from github. Load all the packages required.

#install.packages("tidyverse")  
#install.packages("lubridate")  
#install.packages("randomForest")  
#install.packages("devtools")  
  
  
library(devtools)

## Loading required package: usethis

#Install the bikerentals package from the gitHub repository "Reproduce" of "idahans"  
install\_github("idahans/Reproduce")

## Skipping install of 'bikerentals' from a github remote, the SHA1 (a3c6c966) has not changed since last install.  
## Use `force = TRUE` to force installation

library(lubridate)

##   
## Attaching package: 'lubridate'

## The following objects are masked from 'package:base':  
##   
## date, intersect, setdiff, union

library(tidyverse)

## -- Attaching packages -------------------------------------------------------- tidyverse 1.3.0 --

## v ggplot2 3.3.2 v purrr 0.3.4  
## v tibble 3.0.1 v dplyr 1.0.0  
## v tidyr 1.1.0 v stringr 1.4.0  
## v readr 1.3.1 v forcats 0.5.0

## -- Conflicts ----------------------------------------------------------- tidyverse\_conflicts() --  
## x lubridate::as.difftime() masks base::as.difftime()  
## x lubridate::date() masks base::date()  
## x dplyr::filter() masks stats::filter()  
## x lubridate::intersect() masks base::intersect()  
## x dplyr::lag() masks stats::lag()  
## x lubridate::setdiff() masks base::setdiff()  
## x lubridate::union() masks base::union()

library(randomForest)

## randomForest 4.6-14

## Type rfNews() to see new features/changes/bug fixes.

##   
## Attaching package: 'randomForest'

## The following object is masked from 'package:dplyr':  
##   
## combine

## The following object is masked from 'package:ggplot2':  
##   
## margin

library(bikerentals)

#### Look at the data

The *bikes* data comes with the packages *bikerentals*. Start with how the data set looks like.

head(bikes)

## datetime season holiday workingday weather temp atemp humidity  
## 1 2011-01-01 00:00:00 1 0 0 1 9.84 14.395 81  
## 2 2011-01-01 01:00:00 1 0 0 1 9.02 13.635 80  
## 3 2011-01-01 02:00:00 1 0 0 1 9.02 13.635 80  
## 4 2011-01-01 03:00:00 1 0 0 1 9.84 14.395 75  
## 5 2011-01-01 04:00:00 1 0 0 1 9.84 14.395 75  
## 6 2011-01-01 05:00:00 1 0 0 2 9.84 12.880 75  
## windspeed casual registered count  
## 1 0.0000 3 13 16  
## 2 0.0000 8 32 40  
## 3 0.0000 5 27 32  
## 4 0.0000 3 10 13  
## 5 0.0000 0 1 1  
## 6 6.0032 0 1 1

Look at the structure of the data set

str(bikes)

## 'data.frame': 10886 obs. of 12 variables:  
## $ datetime : chr "2011-01-01 00:00:00" "2011-01-01 01:00:00" "2011-01-01 02:00:00" "2011-01-01 03:00:00" ...  
## $ season : int 1 1 1 1 1 1 1 1 1 1 ...  
## $ holiday : int 0 0 0 0 0 0 0 0 0 0 ...  
## $ workingday: int 0 0 0 0 0 0 0 0 0 0 ...  
## $ weather : int 1 1 1 1 1 2 1 1 1 1 ...  
## $ temp : num 9.84 9.02 9.02 9.84 9.84 ...  
## $ atemp : num 14.4 13.6 13.6 14.4 14.4 ...  
## $ humidity : int 81 80 80 75 75 75 80 86 75 76 ...  
## $ windspeed : num 0 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 ...  
## $ count : int 16 40 32 13 1 1 2 3 8 14 ...

#### Set variables as factors

Set the variables *season*, *holiday*, *workingday* and *weather* as factors.

bikes$season <- as.factor(bikes$season)  
bikes$holiday <- as.factor(bikes$holiday)  
bikes$workingday <- as.factor(bikes$workingday)  
bikes$weather <- as.factor(bikes$weather)

# Results

Start look at the summary of the data set and see if there are any missing values (NAs) in the data.

summary(bikes)

## datetime season holiday workingday weather temp   
## Length:10886 1:2686 0:10575 0:3474 1:7192 Min. : 0.82   
## Class :character 2:2733 1: 311 1:7412 2:2834 1st Qu.:13.94   
## Mode :character 3:2733 3: 859 Median :20.50   
## 4:2734 4: 1 Mean :20.23   
## 3rd Qu.:26.24   
## Max. :41.00   
## atemp humidity windspeed casual   
## Min. : 0.76 Min. : 0.00 Min. : 0.000 Min. : 0.00   
## 1st Qu.:16.66 1st Qu.: 47.00 1st Qu.: 7.002 1st Qu.: 4.00   
## Median :24.24 Median : 62.00 Median :12.998 Median : 17.00   
## Mean :23.66 Mean : 61.89 Mean :12.799 Mean : 36.02   
## 3rd Qu.:31.06 3rd Qu.: 77.00 3rd Qu.:16.998 3rd Qu.: 49.00   
## Max. :45.45 Max. :100.00 Max. :56.997 Max. :367.00   
## registered count   
## Min. : 0.0 Min. : 1.0   
## 1st Qu.: 36.0 1st Qu.: 42.0   
## Median :118.0 Median :145.0   
## Mean :155.6 Mean :191.6   
## 3rd Qu.:222.0 3rd Qu.:284.0   
## Max. :886.0 Max. :977.0

There is no missing values in the data.

#### Extract Variables

Here we use the extract\_var\_function from the *bikerentals* packages.  
Extract the *datetime* column into new columns of *hour*, *day* and *month*, and set these as factors. Extract only the essential variables to a new data frame called *variables*. We are not using the information from *causal* and *registered*, so these columns will be excluded here. Since we have the *datetime* as *hour*, *day* and *month* now, we also exclude the column *datetime*.

variables <- extract\_var(bikes)  
head(variables)

## season holiday workingday weather temp atemp humidity windspeed day hour  
## 1 1 0 0 1 9.84 14.395 81 0.0000 7 0  
## 2 1 0 0 1 9.02 13.635 80 0.0000 7 1  
## 3 1 0 0 1 9.02 13.635 80 0.0000 7 2  
## 4 1 0 0 1 9.84 14.395 75 0.0000 7 3  
## 5 1 0 0 1 9.84 14.395 75 0.0000 7 4  
## 6 1 0 0 2 9.84 12.880 75 6.0032 7 5  
## month  
## 1 1  
## 2 1  
## 3 1  
## 4 1  
## 5 1  
## 6 1

#### Extract Variables and keep count in the data

Here we use the keep\_all\_function from the *bikerentals* package. This function does the same as ectract\_var\_function but also includes the count column. We use this function to make a new data set called *bikes\_all*.

bikes\_all <- keep\_all(bikes)  
head(bikes\_all)

## season holiday workingday weather temp atemp humidity windspeed day hour  
## 1 1 0 0 1 9.84 14.395 81 0.0000 7 0  
## 2 1 0 0 1 9.02 13.635 80 0.0000 7 1  
## 3 1 0 0 1 9.02 13.635 80 0.0000 7 2  
## 4 1 0 0 1 9.84 14.395 75 0.0000 7 3  
## 5 1 0 0 1 9.84 14.395 75 0.0000 7 4  
## 6 1 0 0 2 9.84 12.880 75 6.0032 7 5  
## month count  
## 1 1 16  
## 2 1 40  
## 3 1 32  
## 4 1 13  
## 5 1 1  
## 6 1 1

Now we can look at the structure of *bikes\_all* and see that wee also have *hour*, *day* and *month* as factors.

str(bikes\_all)

## 'data.frame': 10886 obs. of 12 variables:  
## $ season : Factor w/ 4 levels "1","2","3","4": 1 1 1 1 1 1 1 1 1 1 ...  
## $ holiday : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...  
## $ workingday: Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...  
## $ weather : Factor w/ 4 levels "1","2","3","4": 1 1 1 1 1 2 1 1 1 1 ...  
## $ temp : num 9.84 9.02 9.02 9.84 9.84 ...  
## $ atemp : num 14.4 13.6 13.6 14.4 14.4 ...  
## $ humidity : int 81 80 80 75 75 75 80 86 75 76 ...  
## $ windspeed : num 0 0 0 0 0 ...  
## $ day : Factor w/ 7 levels "1","2","3","4",..: 7 7 7 7 7 7 7 7 7 7 ...  
## $ hour : Factor w/ 24 levels "0","1","2","3",..: 1 2 3 4 5 6 7 8 9 10 ...  
## $ month : Factor w/ 12 levels "1","2","3","4",..: 1 1 1 1 1 1 1 1 1 1 ...  
## $ count : int 16 40 32 13 1 1 2 3 8 14 ...

### Random Forest

Now we want to use the Random Forest model to compute the Variable Important Measure and find a ranking of variables that seems to be of importance.

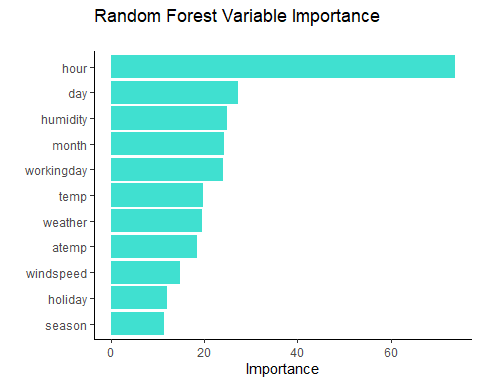
ranfor <- randomForest(variables, bikes$count, ntree = 100, importance = TRUE)  
  
imp <- importance(ranfor, type = 1)  
  
variableImportance <- data.frame(Variable = row.names(imp), Importance = imp[,1])  
  
variableImportance

## Variable Importance  
## season season 11.37642  
## holiday holiday 11.93591  
## workingday workingday 23.97371  
## weather weather 19.40632  
## temp temp 19.68733  
## atemp atemp 18.48289  
## humidity humidity 24.77029  
## windspeed windspeed 14.83727  
## day day 27.18358  
## hour hour 73.83382  
## month month 24.14367

Since Random Forests are stochastic by nature, the results here may change slightly from run to run. However, the results here will show that *hour* have the highest level of importance. We can make a plot of this to get a better overview.

### Plot Random Forest Variable Importance

ggplot(variableImportance, aes(x=reorder(Variable, Importance), y=Importance)) +  
 geom\_bar(stat="identity", fill="turquoise") +  
 coord\_flip() +  
 xlab("") +  
 ylab("Importance") +   
 ggtitle("Random Forest Variable Importance\n") +  
 theme\_classic()

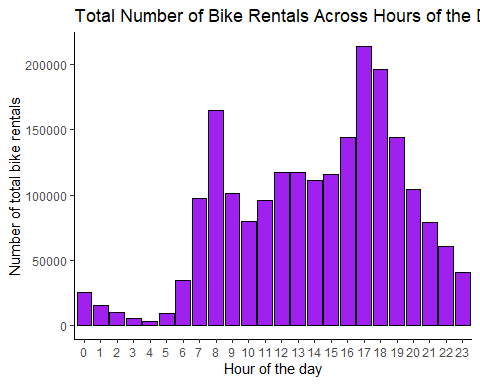


Since the hour of the day seems to be of importance to the demand of bike rentals we can make a plot of total number of bike rentals and hour of the day.

### Plot Bike Rentals and Hour of the day

bikes\_all %>% group\_by(hour) %>%   
 summarize(total = sum(count)) %>%   
 ggplot() +   
 geom\_bar(aes(hour, total), stat = "identity", color = "black", fill = "purple") +   
 ggtitle("Total Number of Bike Rentals Across Hours of the Day") +   
 xlab("Hour of the day") + ylab("Number of total bike rentals") +   
 theme\_classic()

## `summarise()` ungrouping output (override with `.groups` argument)



In this plot we can see the distribution of bike rentals during the hours of the day, and it seems that most bikes are rent at 8 O’clock in the morning and at 5 and 6 in the evening.

# Literature

Fanaee-T, Hadi, and Gama, Joao, Event labeling combining ensemble detectors and background knowledge, Progress in Artificial Intelligence (2013): pp. 1-15, Springer Berlin Heidelberg.

Breiman and Cutler’s Random Forests for Classification and Regression <https://www.stat.berkeley.edu/~breiman/RandomForests/>