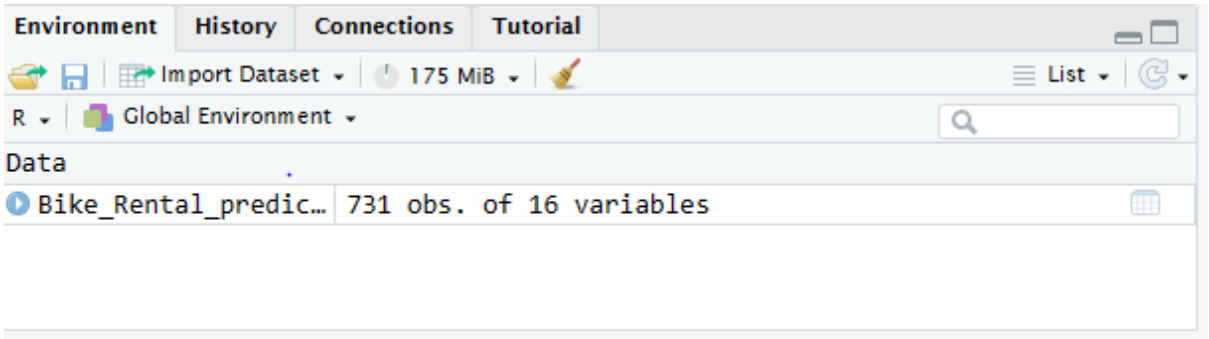
Bike Rental Prediction Project

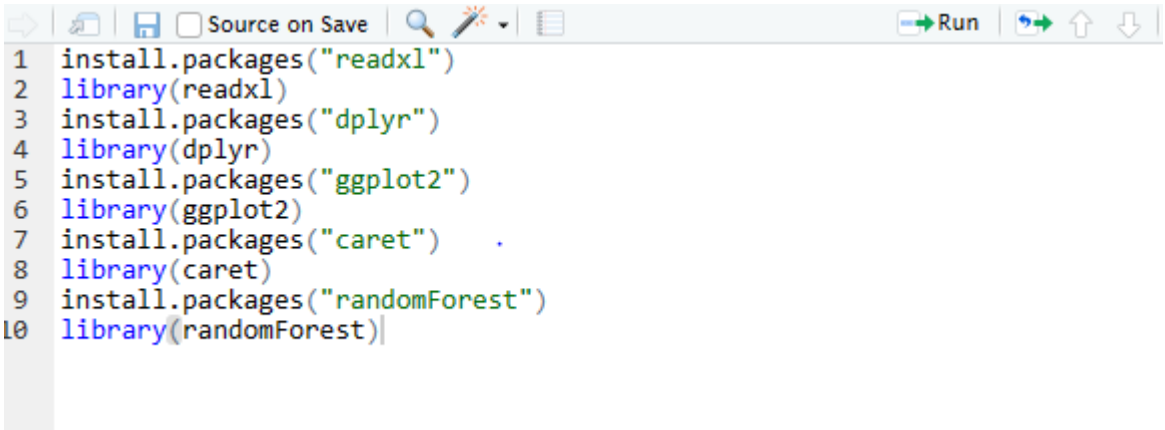
1. Exploratory data analysis

a) Load dataset and libraries

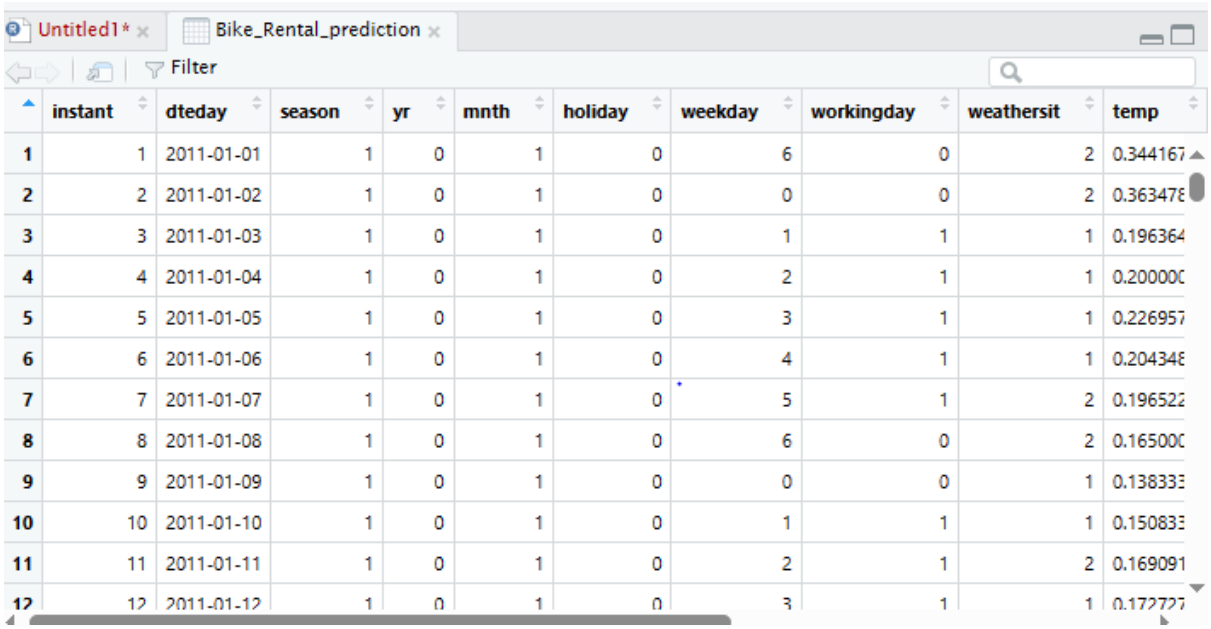
```
# Step 1. a) i. Load Dataset
#--------------------------
setwd(choose.dir())
Bike_Rental_Data_1 <-
read_excel("BikeRentalPredictionDataset.xlsx")
```



Load Libraries



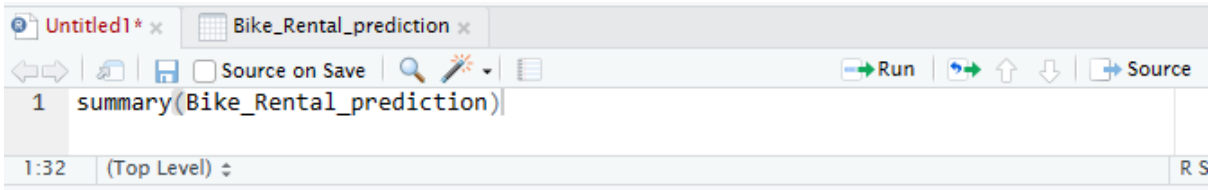
View (Bike_Rental_prediction)



Untitled1* | Bike_Rental_prediction

Filter

	instant	dteday	season	yr	mnth	holiday	weekday	workingday	weathersit	temp
1	1	2011-01-01	1	0	1	0	6	0	2	0.344167
2	2	2011-01-02	1	0	1	0	0	0	2	0.363478
3	3	2011-01-03	1	0	1	0	1	1	1	0.196364
4	4	2011-01-04	1	0	1	0	2	1	1	0.200000
5	5	2011-01-05	1	0	1	0	3	1	1	0.226957
6	6	2011-01-06	1	0	1	0	4	1	1	0.204348
7	7	2011-01-07	1	0	1	0	5	1	2	0.196522
8	8	2011-01-08	1	0	1	0	6	0	2	0.165000
9	9	2011-01-09	1	0	1	0	0	0	1	0.138333
10	10	2011-01-10	1	0	1	0	1	1	1	0.150833
11	11	2011-01-11	1	0	1	0	2	1	2	0.169091
12	12	2011-01-12	1	0	1	0	3	1	1	0.172727




```
> summary(Bike_Rental_prediction)
    instant          dteday                       season           yr
 Min.   :  1.0   Min.   :2011-01-01 00:00:00   Min.   :1.000   Min.   :0.0000
 1st Qu.:183.5   1st Qu.:2011-07-02 12:00:00   1st Qu.:2.000   1st Qu.:0.0000
 Median :366.0   Median :2012-01-01 00:00:00   Median :3.000   Median :1.0000
 Mean   :366.0   Mean   :2012-01-01 00:00:00   Mean   :2.497   Mean   :0.5007
 3rd Qu.:548.5   3rd Qu.:2012-07-01 12:00:00   3rd Qu.:3.000   3rd Qu.:1.0000
 Max.   :731.0   Max.   :2012-12-31 00:00:00   Max.   :4.000   Max.   :1.0000
      mnth          holiday           weekday        workingday      weathersit
 Min.   : 1.00   Min.   :0.00000   Min.   :0.000   Min.   :0.000   Min.   :1.000
 1st Qu.: 4.00   1st Qu.:0.00000   1st Qu.:1.000   1st Qu.:0.000   1st Qu.:1.000
 Median : 7.00   Median :0.00000   Median :3.000   Median :1.000   Median :1.000
 Mean   : 6.52   Mean   :0.02873   Mean   :2.997   Mean   :0.684   Mean   :1.395
 3rd Qu.:10.00   3rd Qu.:0.00000   3rd Qu.:5.000   3rd Qu.:1.000   3rd Qu.:2.000
 Max.   :12.00   Max.   :1.00000   Max.   :6.000   Max.   :1.000   Max.   :3.000
      temp             atemp              hum          windspeed
 Min.   :0.05913   Min.   :0.07907   Min.   :0.0000   Min.   :0.02239
 1st Qu.:0.33708   1st Qu.:0.33784   1st Qu.:0.5200   1st Qu.:0.13495
 Median :0.49833   Median :0.48673   Median :0.6267   Median :0.18097
 Mean   :0.49538   Mean   :0.47435   Mean   :0.6279   Mean   :0.19049
 3rd Qu.:0.65542   3rd Qu.:0.60860   3rd Qu.:0.7302   3rd Qu.:0.23321
 Max.   :0.86167   Max.   :0.84090   Max.   :0.9725   Max.   :0.50746
     casual        registered       cnt
 Min.   :   2.0   Min.   :  20   Min.   :  22
 1st Qu.: 315.5   1st Qu.:2497   1st Qu.:3152
 Median : 713.0   Median :3662   Median :4548
 Mean   : 848.2   Mean   :3656   Mean   :4504
 3rd Qu.:1096.0   3rd Qu.:4776   3rd Qu.:5956
 Max.   :3410.0   Max.   :6946   Max.   :8714
```

```
> str(Bike_Rental_prediction)
tibble [731 × 16] (S3: tbl_df/tbl/data.frame)
 $ instant   : num [1:731] 1 2 3 4 5 6 7 8 9 10 ...
 $ dteday    : POSIXct[1:731], format: "2011-01-01" "2011-01-02" ...
 $ season    : num [1:731] 1 1 1 1 1 1 1 1 1 1 ...
 $ yr        : num [1:731] 0 0 0 0 0 0 0 0 0 0 ...
 $ mnth      : num [1:731] 1 1 1 1 1 1 1 1 1 1 ...
 $ holiday   : num [1:731] 0 0 0 0 0 0 0 0 0 0 ...
 $ weekday   : num [1:731] 6 0 1 2 3 4 5 6 0 1 ...
 $ workingday: num [1:731] 0 0 1 1 1 1 1 0 0 1 ...
 $ weathersit: num [1:731] 2 2 1 1 1 1 2 2 1 1 ...
 $ temp      : num [1:731] 0.344 0.363 0.196 0.2 0.227 ...
 $ atemp     : num [1:731] 0.364 0.354 0.189 0.212 0.229 ...
 $ hum       : num [1:731] 0.806 0.696 0.437 0.59 0.437 ...
 $ windspeed : num [1:731] 0.16 0.249 0.248 0.16 0.187 ...
 $ casual    : num [1:731] 331 131 120 108 82 88 148 68 54 41 ...
 $ registered: num [1:731] 654 670 1229 1454 1518 ...
 $ cnt       : num [1:731] 985 801 1349 1562 1600 ...
>
```

b) Perform data type conversion of the attributes

```
mutate(instant = as.integer(instant), # Convert 'instant' to integer
dteday = as.Date(dteday), # Convert 'dteday' to Date
season = as.factor(season), # Convert 'season' to factor
yr = as.factor(yr), # Convert 'yr' to factor
mnth = as.factor(mnth), # Convert 'mnth' to factor
holiday = as.factor(holiday), # Convert 'holiday' to factor
weekday = as.factor(weekday), # Convert 'weekday' to factor
workingday = as.factor(workingday), # Convert 'workingday' to factor
weathersit = as.factor(weathersit)
)

str(Bike_Rental_prediction)
```

```
:1  (Top Level)                                                    R Script
nsole  Terminal  Jobs
R 4.0.2 . ~/project/
bject 'instant' not found
str(Bike_Rental_prediction)
bble [731 × 16] (S3: tbl_df/tbl/data.frame)
 instant   : int [1:731] 1 2 3 4 5 6 7 8 9 10 ...
 dteday    : Date[1:731], format: "2011-01-01" "2011-01-02" ...
 season    : Factor w/ 4 levels "1","2","3","4": 1 1 1 1 1 1 1 1 1 1 ...
 yr        : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
 mnth      : Factor w/ 12 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 ...
 weekday   : Factor w/ 7 levels "0","1","2","3",..: 7 1 2 3 4 5 6 7 1 2 ...
 workingday: Factor w/ 2 levels "0","1": 1 1 2 2 2 2 2 1 1 2 ...
 weathersit: Factor w/ 3 levels "1","2","3": 2 2 1 1 1 1 2 2 1 1 ...
 temp      : num [1:731] 0.344 0.363 0.196 0.2 0.227 ...
 atemp     : num [1:731] 0.364 0.354 0.189 0.212 0.229 ...
 hum       : num [1:731] 0.806 0.696 0.437 0.59 0.437 ...
 windspeed : num [1:731] 0.16 0.249 0.248 0.16 0.187 ...
 casual    : num [1:731] 331 131 120 108 82 88 148 68 54 41 ...
 registered: num [1:731] 654 670 1229 1454 1518 ...
 cnt       : num [1:731] 985 801 1349 1562 1600 ...
```

c) Carry out the missing value analysis

```
missing_values <- Bike_Rental_prediction
summarize_all(sum(is.na(.)))
print(missing_values)
```

```
> print(missing_values)
# A tibble: 731 × 16
   instant dteday              season    yr  mnth holiday weekday workingday weathersit
     <dbl> <dttm>                <dbl> <dbl> <dbl>   <dbl>   <dbl>      <dbl>      <dbl>
 1       1 2011-01-01 00:00:00      1     0     1       0       6          0          2
 2       2 2011-01-02 00:00:00      1     0     1       0       0          0          2
 3       3 2011-01-03 00:00:00      1     0     1       0       1          1          1
 4       4 2011-01-04 00:00:00      1     0     1       0       2          1          1
 5       5 2011-01-05 00:00:00      1     0     1       0       3          1          1
 6       6 2011-01-06 00:00:00      1     0     1       0       4          1          1
 7       7 2011-01-07 00:00:00      1     0     1       0       5          1          2
 8       8 2011-01-08 00:00:00      1     0     1       0       6          0          2
 9       9 2011-01-09 00:00:00      1     0     1       0       0          0          1
10      10 2011-01-10 00:00:00      1     0     1       0       1          1          1
# i 721 more rows
# i 7 more variables: temp <dbl>, atemp <dbl>, hum <dbl>, windspeed <dbl>,
```

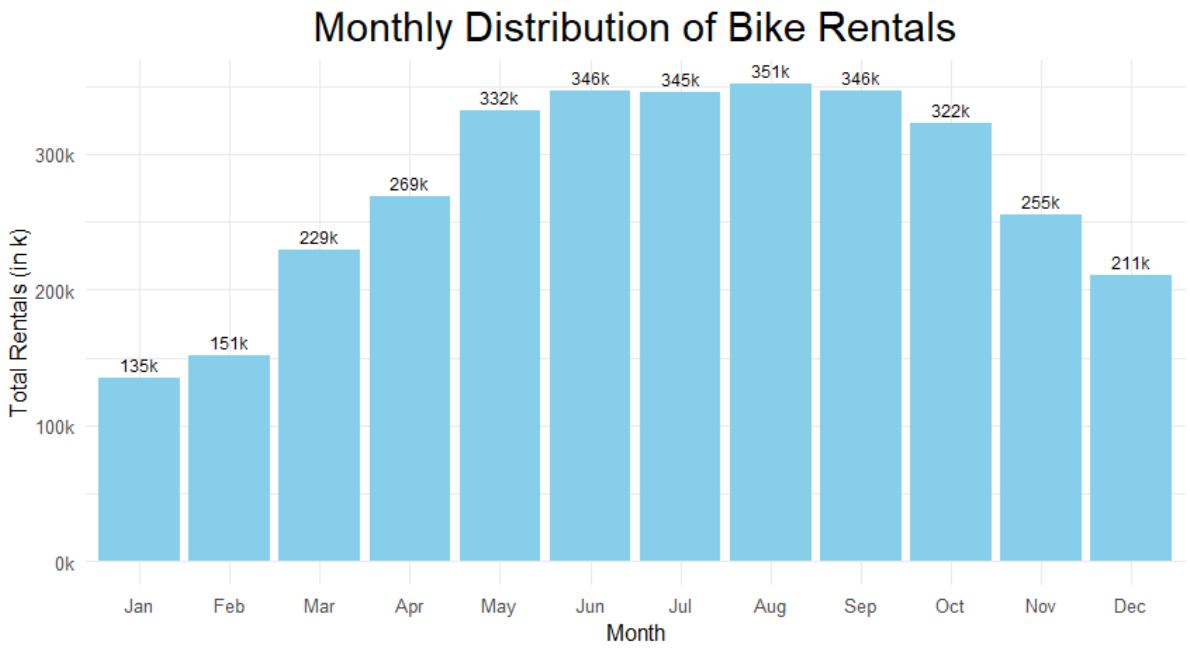
2. Attributes distributions and trends

a) Plot monthly distribution of the total number of bikes rented

```
# Step 2. a) i. Group the data by month and calculate the total number of bikes rented each month
#---------------------------
monthly_rentals <- Bike_Rental_Data_2 %>%
group_by(mnth) %>%
summarise(total_rentals = sum(cnt))
#---------------------------
# Step 2. a) ii. Create a bar plot to visualize the monthly distribution
#---------------------------
ggplot(monthly_rentals, aes(x = mnth, y = total_rentals)) +
geom_bar(stat = "identity", fill = "skyblue") +
geom_text(
aes(label = scales::number_format(scale = 1e-3, accuracy = 1, suffix = "k")(total_rentals)),
vjust = -0.5,
size = 3,
color = "black"
) + # Add labels in thousands (k, without decimals) on top of bars
labs(
title = "Monthly Distribution of Bike Rentals",
x = "Month",
y = "Total Rentals (in k)"
) +
```

```
scale_x_discrete(labels = c(
"1" = "Jan", "2" = "Feb", "3" = "Mar", "4" = "Apr",
"5" = "May", "6" = "Jun", "7" = "Jul", "8" = "Aug",
"9" = "Sep", "10" = "Oct", "11" = "Nov", "12" = "Dec"
)) +
```

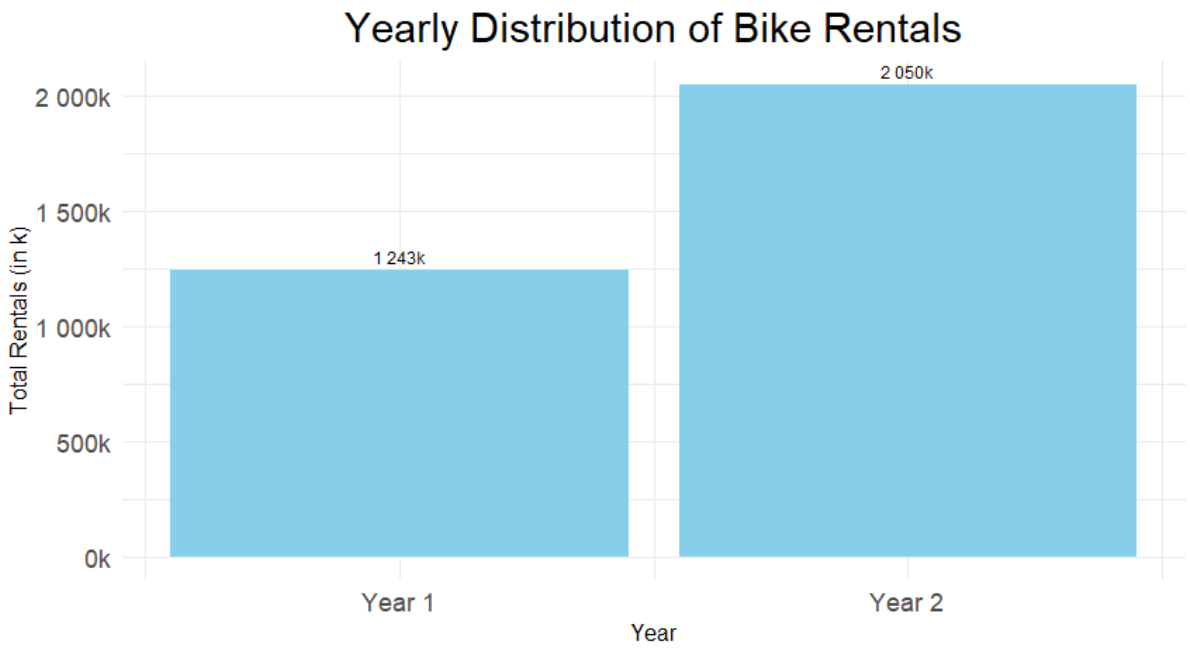
```
scale_y_continuous(
labels = scales::number_format(scale = 1e-3, accuracy = 1, suffix =
"k")
) + # Format Y-axis labels in thousands (k, without decimals) with "k"
suffix
theme_minimal() +
theme(
plot.title = element_text(size = 20, hjust = 0.5) # Adjust size and
center title
)
```



b) Plot yearly distribution of the total number of bikes rented

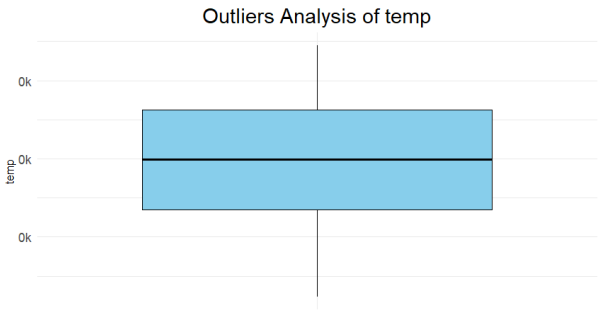
```
# Step 2. b) i. Group the data by year (yr) and calculate the total rentals for each year
#---------------------------
Bike_Rental_Data_2 <- Bike_Rental_Data_2 %>%
mutate(yr = as.numeric(yr))
yearly_rentals <- Bike_Rental_Data_2 %>%
group_by(yr) %>%
summarise(total_rentals = sum(cnt))
#---------------------------
# Step 2. b) ii. Create a bar plot for the yearly distribution of total bike rentals
#---------------------------
ggplot(yearly_rentals, aes(x = yr, y = total_rentals)) +
geom_bar(stat = "identity", fill = "skyblue") +
geom_text(aes(label = scales::number_format(scale = 1e-3, accuracy = 1, suffix = "k")(total_rentals)),
vjust = -0.5, size = 3, color = "black") + # Add labels on top of bars
labs(
title = "Yearly Distribution of Bike Rentals",
x = "Year",
y = "Total Rentals (in k)"
) +
scale_x_continuous(
labels = c("Year 1", "Year 2"), # Specify custom labels
breaks = 1:2 # Specify the breaks for the custom labels
) +
scale_y_continuous(labels = scales::number_format(scale = 1e-3, accuracy = 1, suffix = "k")) +
theme_minimal() +
theme(
plot.title = element_text(size = 20, hjust = 0.5), # Adjust title size and center it
axis.text.x = element_text(size = 12), # Adjust X-axis label font size
axis.text.y = element_text(size = 12) # Adjust Y-axis label font size
```

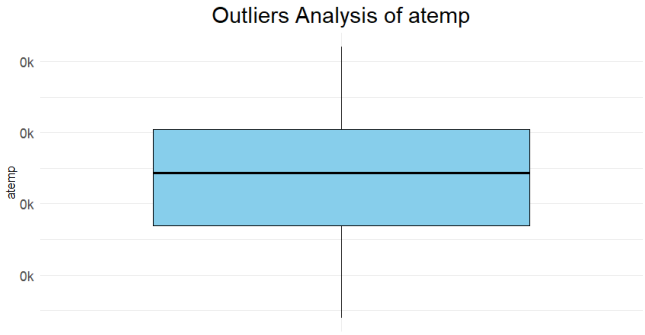
)

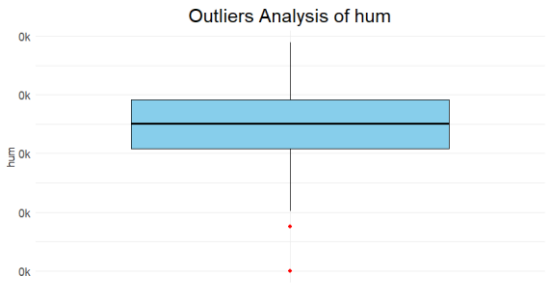

Yearly Distribution of Bike Rentals
2 050k
1 243k
2 000k
1 500k
1 000k
500k
0k
Total Rentals (in k)
Year 1
Year 2
Year

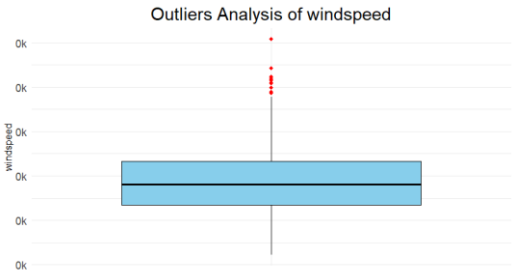
c) Plot boxplot for outliers' analysis

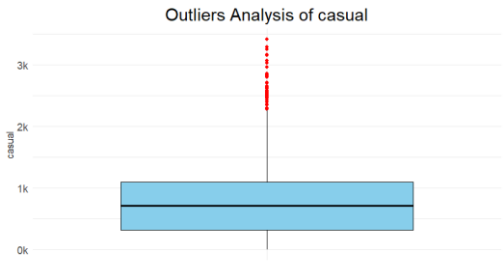
```
# Step 2. c) i. Specify the numeric variables we want to analyse
#---------------------------
numeric_variables <- c("temp", "atemp", "hum", "windspeed",
"casual", "registered", "cnt")
#---------------------------
# Step 2. c) ii. Create a list to store the boxplots
#---------------------------
boxplots <- list()
#---------------------------
# Step 2. c) iii. Create boxplots for each numeric variable
#---------------------------
for (var in numeric_variables) {
p <- ggplot(Bike_Rental_Data_2, aes(x = "", y = !!sym(var))) +
geom_boxplot(fill = "skyblue", color = "black", outlier.color = "red") +
labs(
title = paste("Outliers Analysis of", var),
x = "",
y = var
) +
scale_y_continuous(labels = scales::number_format(scale = 1e-3,
accuracy = 1, suffix = "k")) +
theme_minimal() +
theme(
plot.title = element_text(size = 20, hjust = 0.5), # Adjust title size and
center it
axis.text.x = element_blank(), # Remove X-axis labels
axis.ticks.x = element_blank(), # Remove X-axis ticks
axis.text.y = element_text(size = 12) # Adjust Y-axis label font size
)
boxplots[[var]] <- p
}
boxplots # Print the boxplots
```

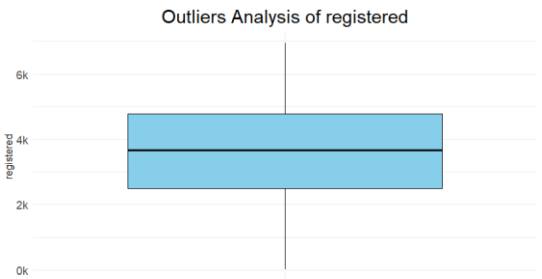

Outliers Analysis of temp
temp
0k
0k
0k

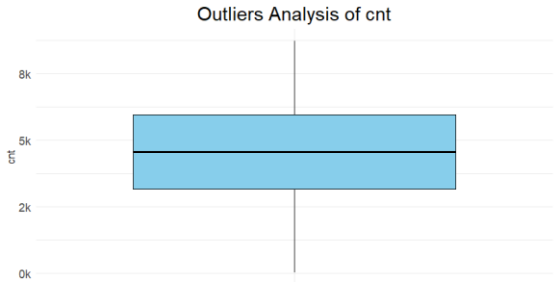

Outliers Analysis of atemp
atemp
0k
0k
0k
0k


Outliers Analysis of hum
hum
0k
0k
0k
0k
0k


Outliers Analysis of windspeed
windspeed
0k
0k
0k
0k
0k
0k


Outliers Analysis of casual
casual
3k
2k
1k
0k


Outliers Analysis of registered
registered
6k
4k
2k
0k


Outliers Analysis of cnt
cnt
8k
5k
2k
0k

3. Split the dataset into train and test dataset

Step 3. a) i. Set a random seed for reproducibility

#----------

set.seed(123)

Step 3. a) ii. Split the data into training (70%) and test (30%) sets

#----------

trainIndex <- createDataPartition(Bike_Rental_Data_2$cnt, p = 0.7, list = FALSE)

training_data <- Bike_Rental_Data_2[trainIndex,]

```
> training_data
# A tibble: 515 × 16
   instant dteday     season    yr mnth  holiday weekday workingday weathersit  temp atemp   hum windspeed
     <int> <date>     <fct>  <dbl> <fct> <fct>   <fct>   <fct>      <fct>      <dbl> <dbl> <dbl>     <dbl>
 1       4 2011-01-04 1          1 1     0       2       1          1          0.2   0.212 0.590     0.160
 2       5 2011-01-05 1          1 1     0       3       1          1          0.227 0.229 0.437     0.187
 3       6 2011-01-06 1          1 1     0       4       1          1          0.204 0.233 0.518     0.0896
 4       7 2011-01-07 1          1 1     0       5       1          2          0.197 0.209 0.499     0.169
 5       8 2011-01-08 1          1 1     0       6       0          2          0.165 0.162 0.536     0.267
 6      11 2011-01-11 1          1 1     0       2       1          2          0.169 0.191 0.686     0.122
 7      12 2011-01-12 1          1 1     0       3       1          1          0.173 0.160 0.600     0.305
 8      13 2011-01-13 1          1 1     0       4       1          1          0.165 0.151 0.470     0.301
 9      14 2011-01-14 1          1 1     0       5       1          1          0.161 0.188 0.538     0.127
10      16 2011-01-16 1          1 1     0       0       0          1          0.232 0.234 0.484     0.188
# i 505 more rows
# i 3 more variables: casual <dbl>, registered <dbl>, cnt <dbl>
# i Use `print(n = ...)` to see more rows
>
```

test_data <- Bike_Rental_Data_2[-trainIndex,]

```
> test_data
# A tibble: 216 × 16
   instant dteday     season    yr mnth  holiday weekday workingday weathersit  temp atemp   hum windspeed
     <int> <date>     <fct>  <dbl> <fct> <fct>   <fct>   <fct>      <fct>      <dbl> <dbl> <dbl>     <dbl>
 1       1 2011-01-01 1          1 1     0       6       0          2          0.344 0.364 0.806     0.160
 2       2 2011-01-02 1          1 1     0       0       0          2          0.363 0.354 0.696     0.249
 3       3 2011-01-03 1          1 1     0       1       1          1          0.196 0.189 0.437     0.248
 4       9 2011-01-09 1          1 1     0       0       0          1          0.138 0.116 0.434     0.362
 5      10 2011-01-10 1          1 1     0       1       1          1          0.151 0.151 0.483     0.223
 6      15 2011-01-15 1          1 1     0       6       0          2          0.233 0.248 0.499     0.158
 7      18 2011-01-18 1          1 1     0       2       1          2          0.217 0.232 0.862     0.147
 8      20 2011-01-20 1          1 1     0       4       1          2          0.262 0.255 0.538     0.196
 9      28 2011-01-28 1          1 1     0       5       1          2          0.203 0.223 0.793     0.123
10      29 2011-01-29 1          1 1     0       6       0          1          0.197 0.212 0.652     0.145
# i 206 more rows
# i 3 more variables: casual <dbl>, registered <dbl>, cnt <dbl>
# i Use `print(n = ...)` to see more rows
>
```

4. Create a model using the random forest algorithm

Step 4. a) i. Train a random forest model

#---------- model <- randomForest(cnt ~ season + yr + mnth + holiday + weekday + workingday + weathersit + temp + atemp + hum + windspeed + casual + registered,

data = training_data)
#---------------------------
Step 4. a) ii. Make predictions on the test data
#---------------------------
predictions <- predict(model, newdata = test_data)

```
> model

Call:
 randomForest(formula = cnt ~ season + yr + mnth + holiday + weekday +      workingday + weathersit + temp + atemp
+ hum + windspeed +      casual + registered, data = training_data)
               Type of random forest: regression
                     Number of trees: 500
No. of variables tried at each split: 4

          Mean of squared residuals: 79088.16
                    % Var explained: 97.89
> |
```

5. Predict the performance of the model on the test dataset

Step 5. a) i. Make predictions on the test data using the trained model
#---------------------------
test_predictions <- predict(model, newdata = test_data)
#---------------------------
Step 5. a) ii. Calculate RMSE (Root Mean Squared Error)
#---------------------------
rmse <- sqrt(mean((test_data$cnt - test_predictions)^2))
cat("Root Mean Squared Error (RMSE):", rmse, "\n")

```
> rmse <- sqrt(mean((test_data$cnt - test_predictions)^2))
> rmse
[1] 300.5998
> cat("Root Mean Squared Error (RMSE):", rmse, "\n")
Root Mean Squared Error (RMSE): 300.5998
> |
```

Step 5. a) iii. Calculate R-squared (coefficient of determination) to measure the goodness of fit
#---------------------------

# R-squared measures the proportion of the variance in the dependent variable (cnt) that is predictable from the independent variables.
r_squared <- 1 - (sum((test_data$cnt - test_predictions)^2) / sum((test_data$cnt - mean(test_data$cnt))^2))
cat("R-squared (R2):", r_squared, "\n")

```
> r_squared <- 1 - (sum((test_data$cnt - test_predictions)^2) / sum((test_data$cnt - mean(test_data$cnt))^2))
> cat("R-squared (R2):", r_squared, "\n")
R-squared (R2): 0.9758788
>
```

Step 5. a) iv. Create a scatterplot of actual vs. predicted values
#---------------------------
plot(test_data$cnt, test_predictions, xlab = "Actual", ylab = "Predicted", main = "Actual vs. Predicted Values")
abline(0, 1, col = "red") # Add a diagonal line for reference