Weather and Citibike Usage

Charley Ferrari

Thursday, December 11, 2014

```
#Packages used:

library(rvest)
library(dplyr)
library(plyr)
library(zoo)
library(RPostgreSQL)
library(ggmap)
library(ggplot2)
library(ggthemes)
```

Introduction

The Citibike bikeshare system in New York City produces detailed ride level data for research purposes. This data details each trip taken by a citibike, including information about the bike, the rider, timing, and the start and end station. Inspired by the Kaggle competition's bike share challenge, which included aggregate hourly data joined with weather data, I decided to rebuild this framework using the raw rides data coming from Citibike.

The weather data was scraped directly from the WUnderground website, looping through daily pages. Weekends are expressed as a dummy variable, on an hourly basis. The citibike data was just obtained from the citibike site as raw CSVs

For analysis, I created a regression model that predicts hourly Citibike usage based on weather and weekend. Because I had the trip level data, I also created a map with station aggregated data animated by hour, to get a look at an average day's flow of bikes. This data is divided into both a weekday and weekend average.

Data was brought into PostgreSQL as soon as possible. The majority of aggregations were done in PostgreSQL, and tables were then imported into R.

Data Profile

Weather data:

Weather data was scraped using the following code. The for loops were complicated due to the unclean nature of the weather data, and the entire code is reproduced in Appendix A. The loops recreate the html addresses, an example of which can be found here. The table at the bottom of the page was scraped.

```
monthlookup <- data.frame(
  monthnum = 1:12,
  numdays = c(31,28,31,30,31,30,31,31,30,31,30,31))


df <- data.frame(year = numeric(0),
                 month = numeric(0),
                 day = numeric(0),
                 hour = numeric(0),
                 minute = numeric(0),
```

```
                  temp = numeric(0),
                  windchill = numeric(0),
                  heatindex = numeric(0),
                  dewpoint = numeric(0),
                  humidity = numeric(0),
                  pressure = numeric(0),
                  visibility = numeric(0),
                  winddir = character(0),
                  windspeed = numeric(0),
                  gustspeed = numeric(0),
                  precip = numeric(0),
                  events = character(0),
                  conditions = character(0),
                  stringsAsFactors = FALSE)


for(monthindex in 7:12){


for(date in 1:filter(monthlookup, monthnum == monthindex)$numdays){

  weatherpage <- html(paste("http://www.wunderground.com/history/airport/KNYC/2013/",monthindex,"/",date
                       "/DailyHistory.html", sep = ""))

  #####
  # The columns change. The third column could either be Windchill, Heat Index
  # Or Dew Point (which should be the fourth column)
  #####

  weathertest <- weatherpage %>%
    html_nodes("#obsTable > thead > tr > th:nth-child(3)") %>%
    html_text()

  time <- weatherpage %>%
    html_nodes("#obsTable > tbody > tr > td:nth-child(1)") %>%
    html_text()
  time <- paste(monthindex,"/",date,"/2013 ", time, sep="")
  time <- strptime(time, format="%m/%d/%Y %I:%M %p")

  year <- time$year+1900
  month <- time$mon+1
  day <- time$mday
  hour <- time$hour
  minute <- time$min

  temp <- weatherpage %>%
    html_nodes("#obsTable > tbody > tr > td:nth-child(2) > span > span.wx-value") %>%
    html_text() %>%
    as.numeric()

  if(weathertest == "Windchill"){

      windchill <- weatherpage %>%
```

```
      html_nodes("#obsTable > tbody > tr > td:nth-child(3)") %>%
      html_text()
    windchill[windchill == "\n  -\n"] <- NA
    windchill[!is.na(windchill)] <- substr(windchill[!is.na(windchill)], 4,
                                           nchar(windchill[!is.na(windchill)])-4)
    windchill <- as.numeric(windchill)

    heatindex <- rep(NA, each=length(temp))

    print("Windchill")

}

if(weathertest == "Heat Index"){

    windchill <- rep(NA, each=length(temp))

    heatindex <- weatherpage %>%
      html_nodes("#obsTable > tbody > tr > td:nth-child(3)") %>%
      html_text()
    heatindex[heatindex == "\n  -\n"] <- NA
    heatindex[!is.na(heatindex)] <- substr(heatindex[!is.na(heatindex)], 4,
                                           nchar(heatindex[!is.na(heatindex)])-4)
    heatindex <- as.numeric(heatindex)

    print("heatindex")
}

if(weathertest == "Dew Point"){

    windchill <- rep(NA, each=length(temp))

    heatindex <- rep(NA, each=length(temp))

    print("dew point")
}


if(weathertest == "Windchill" | weathertest == "Heat Index"){

    dewpoint <- weatherpage %>%
      html_nodes("#obsTable > tbody > tr > td:nth-child(4) > span > span.wx-value") %>%
      html_text() %>%
      as.numeric()

    humidity <- weatherpage %>%
      html_nodes("#obsTable > tbody > tr > td:nth-child(5)") %>%
      html_text()
    humidity <- as.numeric(substr(humidity, 1, 2))



    #pressure <- weatherpage %>%
```

```
#  html_nodes("#obsTable > tbody > tr > td:nth-child(6) > span > span.wx-value") %>%
#  html_text() %>%
#  as.numeric()

pressure <- weatherpage %>%
  html_nodes("#obsTable > tbody > tr > td:nth-child(6)") %>%
  html_text()
pressure[pressure == "\n  -\n"] <- NA
pressure[!is.na(pressure)] <- substr(pressure[!is.na(pressure)],4,
                                     nchar(pressure[!is.na(pressure)])-4)
pressure <- as.numeric(pressure)

visibility <- weatherpage %>%
  html_nodes("#obsTable > tbody > tr > td:nth-child(7)") %>%
  html_text()
visibility[visibility == "\n  -\n"] <- NA
visibility[!is.na(visibility)] <- substr(visibility[!is.na(visibility)],4,
                                     nchar(visibility[!is.na(visibility)])-4)
visibility <- as.numeric(visibility)

winddir <- weatherpage %>%
  html_nodes("#obsTable > tbody > tr > td:nth-child(8)") %>%
  html_text()

windspeed <- weatherpage %>%
  html_nodes("#obsTable > tbody > tr > td:nth-child(9)") %>%
  html_text()
windspeed[windspeed == "\n  -\n"] <- NA
windspeed[windspeed == "Calm"] <- NA
windspeed[!is.na(windspeed)] <- substr(windspeed[!is.na(windspeed)], 4,
         nchar(windspeed[!is.na(windspeed)])-5)
windspeed <- as.numeric(windspeed)

#heatindex <- rep(NA, each=length(windspeed))

gustspeed <- weatherpage %>%
  html_nodes("#obsTable > tbody > tr > td:nth-child(10)") %>%
  html_text()
gustspeed[gustspeed == "\n  -\n"] <- NA
gustspeed[!is.na(gustspeed)] <- as.numeric(substr(gustspeed[!is.na(gustspeed)],4,7))

precip <- weatherpage %>%
  html_nodes("#obsTable > tbody > tr > td:nth-child(11)") %>%
  html_text()
precip[precip == "N/A"] <- NA
precip[!is.na(precip)] <- as.numeric(substr(precip[!is.na(precip)],4,7))

eventTestPage <- html("http://www.wunderground.com/history/airport/KNYC/2014/12/3/DailyHistory.htm
eventTest <- eventTestPage %>%
  html_nodes("#obsTable > tbody > tr:nth-child(1) > td:nth-child(12)") %>%
  html_text()

events <- weatherpage %>%
```

```
      html_nodes("#obsTable > tbody > tr > td:nth-child(12)") %>%
      html_text()
    events[events == eventTest] <- NA
    events[!is.na(events)] <- substr(events[!is.na(events)], 2, nchar(events[!is.na(events)])-1)

    conditions <- weatherpage %>%
      html_nodes("#obsTable > tbody > tr > td:nth-child(13)") %>%
      html_text()

}

if(weathertest == "Dew Point"){

  dewpoint <- weatherpage %>%
    html_nodes("#obsTable > tbody > tr > td:nth-child(3) > span > span.wx-value") %>%
    html_text() %>%
    as.numeric()

  humidity <- weatherpage %>%
    html_nodes("#obsTable > tbody > tr > td:nth-child(4)") %>%
    html_text()
  humidity <- as.numeric(substr(humidity, 1, 2))

  pressure <- weatherpage %>%
    html_nodes("#obsTable > tbody > tr > td:nth-child(5)") %>%
    html_text()
  pressure[pressure == "\n  -\n"] <- NA
  pressure[!is.na(pressure)] <- substr(pressure[!is.na(pressure)],4,
                                       nchar(pressure[!is.na(pressure)])-4)
  pressure <- as.numeric(pressure)

  visibility <- weatherpage %>%
    html_nodes("#obsTable > tbody > tr > td:nth-child(6)") %>%
    html_text()
  visibility[visibility == "\n  -\n"] <- NA
  visibility[!is.na(visibility)] <- substr(visibility[!is.na(visibility)],4,
                                       nchar(visibility[!is.na(visibility)])-4)
  visibility <- as.numeric(visibility)

  winddir <- weatherpage %>%
    html_nodes("#obsTable > tbody > tr > td:nth-child(7)") %>%
    html_text()

  windspeed <- weatherpage %>%
    html_nodes("#obsTable > tbody > tr > td:nth-child(8)") %>%
    html_text()
  windspeed[windspeed == "\n  -\n"] <- NA
  windspeed[windspeed == "Calm"] <- NA
  windspeed[!is.na(windspeed)] <- substr(windspeed[!is.na(windspeed)], 4,
                                         nchar(windspeed[!is.na(windspeed)])-5)
  windspeed <- as.numeric(windspeed)

  heatindex <- rep(NA, each=length(windspeed))
```

```
    gustspeed <- weatherpage %>%
      html_nodes("#obsTable > tbody > tr > td:nth-child(9)") %>%
      html_text()
    gustspeed[gustspeed == "\n  -\n"] <- NA
    gustspeed[!is.na(gustspeed)] <- as.numeric(substr(gustspeed[!is.na(gustspeed)],4,7))

    precip <- weatherpage %>%
      html_nodes("#obsTable > tbody > tr > td:nth-child(10)") %>%
      html_text()
    precip[precip == "N/A"] <- NA
    precip[!is.na(precip)] <- as.numeric(substr(precip[!is.na(precip)],4,7))

    eventTestPage <- html("http://www.wunderground.com/history/airport/KNYC/2014/12/3/DailyHistory.html
    eventTest <- eventTestPage %>%
      html_nodes("#obsTable > tbody > tr:nth-child(1) > td:nth-child(12)") %>%
      html_text()

    events <- weatherpage %>%
      html_nodes("#obsTable > tbody > tr > td:nth-child(11)") %>%
      html_text()
    events[events == eventTest] <- NA
    events[!is.na(events)] <- substr(events[!is.na(events)], 2, nchar(events[!is.na(events)])-1)

    conditions <- weatherpage %>%
      html_nodes("#obsTable > tbody > tr > td:nth-child(12)") %>%
      html_text()

  }

  dfadd <- data.frame(cbind(
    year, month, day, hour, minute, temp, windchill, heatindex, dewpoint,
    humidity,pressure, visibility, winddir, windspeed, gustspeed, precip,
    events, conditions), stringsAsFactors = FALSE)

  df <- rbind( df,dfadd)

  rm(dfadd)


}



}

for(monthindex in 1:8){


  for(date in 1:filter(monthlookup, monthnum == monthindex)$numdays){

    weatherpage <- html(paste("http://www.wunderground.com/history/airport/KNYC/2014/",monthindex,"/",da
                              "/DailyHistory.html", sep = ""))
```

```
#####
# The columns change. The third column could either be Windchill, Heat Index
# Or Dew Point (which should be the fourth column)
#####

weathertest <- weatherpage %>%
  html_nodes("#obsTable > thead > tr > th:nth-child(3)") %>%
  html_text()

time <- weatherpage %>%
  html_nodes("#obsTable > tbody > tr > td:nth-child(1)") %>%
  html_text()
time <- paste(monthindex,"/",date,"/2014 ", time, sep="")
time <- strptime(time, format="%m/%d/%Y %I:%M %p")

year <- time$year+1900
month <- time$mon+1
day <- time$mday
hour <- time$hour
minute <- time$min

temp <- weatherpage %>%
  html_nodes("#obsTable > tbody > tr > td:nth-child(2) > span > span.wx-value") %>%
  html_text() %>%
  as.numeric()

if(weathertest == "Windchill"){

  windchill <- weatherpage %>%
    html_nodes("#obsTable > tbody > tr > td:nth-child(3)") %>%
    html_text()
  windchill[windchill == "\n  -\n"] <- NA
  windchill[!is.na(windchill)] <- substr(windchill[!is.na(windchill)], 4,
                                         nchar(windchill[!is.na(windchill)])-4)
  windchill <- as.numeric(windchill)

  heatindex <- rep(NA, each=length(temp))

  print("Windchill")

}

if(weathertest == "Heat Index"){

  windchill <- rep(NA, each=length(temp))

  heatindex <- weatherpage %>%
    html_nodes("#obsTable > tbody > tr > td:nth-child(3)") %>%
    html_text()
  heatindex[heatindex == "\n  -\n"] <- NA
  heatindex[!is.na(heatindex)] <- substr(heatindex[!is.na(heatindex)], 4,
                                         nchar(heatindex[!is.na(heatindex)])-4)
  heatindex <- as.numeric(heatindex)
```

```
  print("heatindex")
}

if(weathertest == "Dew Point"){

  windchill <- rep(NA, each=length(temp))

  heatindex <- rep(NA, each=length(temp))

  print("dew point")
}


if(weathertest == "Windchill" | weathertest == "Heat Index"){

  dewpoint <- weatherpage %>%
    html_nodes("#obsTable > tbody > tr > td:nth-child(4) > span > span.wx-value") %>%
    html_text() %>%
    as.numeric()

  humidity <- weatherpage %>%
    html_nodes("#obsTable > tbody > tr > td:nth-child(5)") %>%
    html_text()
  humidity <- as.numeric(substr(humidity, 1, 2))



  #pressure <- weatherpage %>%
  #  html_nodes("#obsTable > tbody > tr > td:nth-child(6) > span > span.wx-value") %>%
  #  html_text() %>%
  #  as.numeric()

  pressure <- weatherpage %>%
    html_nodes("#obsTable > tbody > tr > td:nth-child(6)") %>%
    html_text()
  pressure[pressure == "\n  -\n"] <- NA
  pressure[!is.na(pressure)] <- substr(pressure[!is.na(pressure)],4,
                                       nchar(pressure[!is.na(pressure)])-4)
  pressure <- as.numeric(pressure)

  visibility <- weatherpage %>%
    html_nodes("#obsTable > tbody > tr > td:nth-child(7)") %>%
    html_text()
  visibility[visibility == "\n  -\n"] <- NA
  visibility[!is.na(visibility)] <- substr(visibility[!is.na(visibility)],4,
                                           nchar(visibility[!is.na(visibility)])-4)
  visibility <- as.numeric(visibility)

  winddir <- weatherpage %>%
    html_nodes("#obsTable > tbody > tr > td:nth-child(8)") %>%
    html_text()

  windspeed <- weatherpage %>%
```

```
    html_nodes("#obsTable > tbody > tr > td:nth-child(9)") %>%
    html_text()
  windspeed[windspeed == "\n  -\n"] <- NA
  windspeed[windspeed == "Calm"] <- NA
  windspeed[!is.na(windspeed)] <- substr(windspeed[!is.na(windspeed)], 4,
                                         nchar(windspeed[!is.na(windspeed)])-5)
  windspeed <- as.numeric(windspeed)

  #heatindex <- rep(NA, each=length(windspeed))

  gustspeed <- weatherpage %>%
    html_nodes("#obsTable > tbody > tr > td:nth-child(10)") %>%
    html_text()
  gustspeed[gustspeed == "\n  -\n"] <- NA
  gustspeed[!is.na(gustspeed)] <- as.numeric(substr(gustspeed[!is.na(gustspeed)],4,7))

  precip <- weatherpage %>%
    html_nodes("#obsTable > tbody > tr > td:nth-child(11)") %>%
    html_text()
  precip[precip == "N/A"] <- NA
  precip[!is.na(precip)] <- as.numeric(substr(precip[!is.na(precip)],4,7))

  eventTestPage <- html("http://www.wunderground.com/history/airport/KNYC/2014/12/3/DailyHistory.htm
  eventTest <- eventTestPage %>%
    html_nodes("#obsTable > tbody > tr:nth-child(1) > td:nth-child(12)") %>%
    html_text()

  events <- weatherpage %>%
    html_nodes("#obsTable > tbody > tr > td:nth-child(12)") %>%
    html_text()
  events[events == eventTest] <- NA
  events[!is.na(events)] <- substr(events[!is.na(events)], 2, nchar(events[!is.na(events)])-1)

  conditions <- weatherpage %>%
    html_nodes("#obsTable > tbody > tr > td:nth-child(13)") %>%
    html_text()

}

if(weathertest == "Dew Point"){

  dewpoint <- weatherpage %>%
    html_nodes("#obsTable > tbody > tr > td:nth-child(3) > span > span.wx-value") %>%
    html_text() %>%
    as.numeric()

  humidity <- weatherpage %>%
    html_nodes("#obsTable > tbody > tr > td:nth-child(4)") %>%
    html_text()
  humidity <- as.numeric(substr(humidity, 1, 2))

  pressure <- weatherpage %>%
    html_nodes("#obsTable > tbody > tr > td:nth-child(5)") %>%
```

```
  html_text()
pressure[pressure == "\n  -\n"] <- NA
pressure[!is.na(pressure)] <- substr(pressure[!is.na(pressure)],4,
                                     nchar(pressure[!is.na(pressure)])-4)
pressure <- as.numeric(pressure)

visibility <- weatherpage %>%
  html_nodes("#obsTable > tbody > tr > td:nth-child(6)") %>%
  html_text()
visibility[visibility == "\n  -\n"] <- NA
visibility[!is.na(visibility)] <- substr(visibility[!is.na(visibility)],4,
                                         nchar(visibility[!is.na(visibility)])-4)
visibility <- as.numeric(visibility)

winddir <- weatherpage %>%
  html_nodes("#obsTable > tbody > tr > td:nth-child(7)") %>%
  html_text()

windspeed <- weatherpage %>%
  html_nodes("#obsTable > tbody > tr > td:nth-child(8)") %>%
  html_text()
windspeed[windspeed == "\n  -\n"] <- NA
windspeed[windspeed == "Calm"] <- NA
windspeed[!is.na(windspeed)] <- substr(windspeed[!is.na(windspeed)], 4,
                                       nchar(windspeed[!is.na(windspeed)])-5)
windspeed <- as.numeric(windspeed)

heatindex <- rep(NA, each=length(windspeed))

gustspeed <- weatherpage %>%
  html_nodes("#obsTable > tbody > tr > td:nth-child(9)") %>%
  html_text()
gustspeed[gustspeed == "\n  -\n"] <- NA
gustspeed[!is.na(gustspeed)] <- as.numeric(substr(gustspeed[!is.na(gustspeed)],4,7))

precip <- weatherpage %>%
  html_nodes("#obsTable > tbody > tr > td:nth-child(10)") %>%
  html_text()
precip[precip == "N/A"] <- NA
precip[!is.na(precip)] <- as.numeric(substr(precip[!is.na(precip)],4,7))

eventTestPage <- html("http://www.wunderground.com/history/airport/KNYC/2014/12/3/DailyHistory.htm
eventTest <- eventTestPage %>%
  html_nodes("#obsTable > tbody > tr:nth-child(1) > td:nth-child(12)") %>%
  html_text()

events <- weatherpage %>%
  html_nodes("#obsTable > tbody > tr > td:nth-child(11)") %>%
  html_text()
events[events == eventTest] <- NA
events[!is.na(events)] <- substr(events[!is.na(events)], 2, nchar(events[!is.na(events)])-1)

conditions <- weatherpage %>%
```

```
        html_nodes("#obsTable > tbody > tr > td:nth-child(12)") %>%
        html_text()

    }

    dfadd <- data.frame(cbind(
      year, month, day, hour, minute, temp, windchill, heatindex, dewpoint,
      humidity,pressure, visibility, winddir, windspeed, gustspeed, precip,
      events, conditions), stringsAsFactors = FALSE)

    df <- rbind( df,dfadd)

    rm(dfadd)


  }



}

df$year <- as.numeric(df$year)
df$month <- as.numeric(df$month)
df$day <- as.numeric(df$day)
df$hour <- as.numeric(df$hour)
df$minute <- as.numeric(df$minute)
df$temp <- as.numeric(df$temp)
df$windchill <- as.numeric(df$windchill)
df$heatindex <- as.numeric(df$heatindex)
df$dewpoint <- as.numeric(df$dewpoint)
df$humidity <- as.numeric(df$humidity)
df$pressure <- as.numeric(df$pressure)
df$visibility <- as.numeric(df$visibility)
df$windspeed <- as.numeric(df$windspeed)
df$gustspeed <- as.numeric(df$gustspeed)
df$precip <- as.numeric(df$precip)
```

Trip Data:

The trip data from the source CSV included the following columns:

- Trip Duration (seconds): num
- Start Time and Date: character, converted to POSIXlt(varchar in PostgreSQL though)
- stop Time and Date: same as above
- Start Station Name: character
- End Station Name: character
- Station ID: num
- Station Latitude: num
- Station Longitude: num
- Bike ID: num
- User Type: character
- Gender: num
- Year of Birth: num

In addition to this, I parsed out the date to get separate columns for years, months, days, hours, minutes, and seconds. Except for minutes and seconds, I'll be using these columns for aggregation purposes.

```
workingdirectory <- "C:/Users/Charley/Downloads/CUNY/IS 607 Data Acquisition and Management/Semester Pr
setwd(workingdirectory)

years2013 <- c("08","09","10","11","12")
years2014 <- c("01","02","03","04","05","06","07","08")
csvs <- paste("2013-",years2013," - Citi Bike trip data.csv",sep="")
csvs <- c(csvs,paste("2014-",years2014," - Citi Bike trip data.csv",sep=""))

bikesharedf <- read.csv("2013-07 - Citi Bike trip data.csv", stringsAsFactors=FALSE)

for(filename in csvs){
  bikesharedf <- rbind(bikesharedf, read.csv(filename))
}

bikesharedf$birth.year[bikesharedf$birth.year == "\\N"] <- NA
bikesharedf$birth.year <- as.numeric(bikesharedf$birth.year)
```

Using this data, I took out the stations and made a separate stations table for my PostgreSQL database. I was then able to remove the station information columns from the trips table. I noticed a few duplicate stations, which seemed to be the result of switching the locations slightly. For these, I just erased the duplicates, as a spot check revealed they were trivially close to their original locations.

```
stationlist <- unique(select(bikesharedf, start.station.id, start.station.name,
                             start.station.latitude, start.station.longitude))
row.names(stationlist) <- NULL
colnames(stationlist) <- c("station.id", "station.name", "station.latitude",
                           "station.longitude")

## Checking the duplicates to make sure they're not too far apart

stationlist %>%
  filter(station.id %in% stationlist$station.id[duplicated(stationlist$station.id)]) %>%
  arrange(station.id)

## Close enough to remove the duplicates

stationlist <- stationlist[!duplicated(stationlist$station.id),]

bikesharedf <- bikesharedf %>%
  select(-(start.station.name:start.station.longitude),
         -(end.station.name:end.station.longitude))

bikesharedf$starttime <- strptime(bikesharedf$starttime,
                                  format = "%Y-%m-%d %H:%M:%S")

bikesharedf$stoptime <- strptime(bikesharedf$stoptime,
                                 format = "%Y-%m-%d %H:%M:%S")



bikesharedf$starthour <- bikesharedf$starttime$hour
```

```
bikesharedf$startday <- bikesharedf$starttime$mday
bikesharedf$startmonth <- bikesharedf$starttime$mon+1
bikesharedf$startyear <- bikesharedf$starttime$year+1900
bikesharedf$startminute <- bikesharedf$starttime$min
bikesharedf$startsecond <- bikesharedf$starttime$sec

bikesharedf$endhour <- bikesharedf$stoptime$hour
bikesharedf$endday <- bikesharedf$stoptime$mday
bikesharedf$endmonth <- bikesharedf$stoptime$mon+1
bikesharedf$endyear <- bikesharedf$stoptime$year+1900
bikesharedf$endminute <- bikesharedf$stoptime$min
bikesharedf$endsecond <- bikesharedf$stoptime$sec
```

I also created a quick daily weekend file, and made the dataframe hourly for use in the hourly comparisons:

```
weekends <- read.csv("weekends.csv")

hours <- 0:23

addhours <- function(vec)
{
  weekends$hours <- vec
  return(weekends)
}

weekends <- adply(hours, .margins = 1, .fun=addhours)

weekends <- weekends %>%
  select(-X1)

colnames(weekends) <- c("date", "day", "month", "year", "weekend", "hour")
```

Methodology

Once the preliminary data was brought into PostgreSQL, tables were created there for further analysis. Below is the R code that brought this data into PostgreSQL:

```
library(RPostgreSQL)

con <- dbConnect(RPostgreSQL::PostgreSQL(), user="postgres", password="insertepasswordhere",
                 dbname="bikeshare")

con

dbWriteTable(con, "hourlyweather", df, row.names=FALSE)

dbWriteTable(con, "trips", bikesharedf, row.names=TRUE)

dbWriteTable(con, "stations", stationlist, row.names=FALSE)

dbWriteTable(con, "weekends", weekends, row.names=FALSE)
```

I kept the row.names TRUE for trips to use as my primary key.

Once in PostgreSQL I created a few primary and foreign keys:

```
# -- Constraint: stations_pkey
#
# -- ALTER TABLE stations DROP CONSTRAINT stations_pkey;
#
# ALTER TABLE stations
# ADD CONSTRAINT stations_pkey PRIMARY KEY("station.id");
#
# -- Constraint: hourlyweather_pkey
#
# -- ALTER TABLE hourlyweather DROP CONSTRAINT hourlyweather_pkey;
#
# ALTER TABLE hourlyweather
#   ADD CONSTRAINT hourlyweather_pkey PRIMARY KEY(year, month, day, hour);
#
# -- Constraint: trips_pkey
#
# -- ALTER TABLE trips DROP CONSTRAINT trips_pkey;
#
# ALTER TABLE trips
#   ADD CONSTRAINT trips_pkey PRIMARY KEY("row.names");
#
# -- Foreign Key: endstation_fkey
#
# -- ALTER TABLE trips DROP CONSTRAINT endstation_fkey;
#
# ALTER TABLE trips
#   ADD CONSTRAINT endstation_fkey FOREIGN KEY ("end.station.id")
#       REFERENCES stations ("station.id") MATCH SIMPLE
#       ON UPDATE NO ACTION ON DELETE NO ACTION;
#
# -- Foreign Key: endweather_fkey
#
# -- ALTER TABLE trips DROP CONSTRAINT endweather_fkey;
#
# ALTER TABLE trips
#   ADD CONSTRAINT endweather_fkey FOREIGN KEY (endyear, endmonth, endday, endhour)
#       REFERENCES hourlyweather (year, month, day, hour) MATCH SIMPLE
#       ON UPDATE NO ACTION ON DELETE NO ACTION;
#
# -- Foreign Key: endweekend_fkey
#
# -- ALTER TABLE trips DROP CONSTRAINT endweekend_fkey;
#
# ALTER TABLE trips
#   ADD CONSTRAINT endweekend_fkey FOREIGN KEY (endyear, endmonth, endday, endhour)
#       REFERENCES weekends (year, month, day, hour) MATCH SIMPLE
#       ON UPDATE NO ACTION ON DELETE NO ACTION;
#
# -- Foreign Key: startstation_fkey
#
# -- ALTER TABLE trips DROP CONSTRAINT startstation_fkey;
```

```
#
# ALTER TABLE trips
#   ADD CONSTRAINT startstation_fkey FOREIGN KEY ("start.station.id")
#       REFERENCES stations ("station.id") MATCH SIMPLE
#       ON UPDATE NO ACTION ON DELETE NO ACTION;
#
#
# -- Foreign Key: startweather_fkey
#
# -- ALTER TABLE trips DROP CONSTRAINT startweather_fkey;
#
# ALTER TABLE trips
#   ADD CONSTRAINT startweather_fkey FOREIGN KEY (startyear, startmonth, startday, starthour)
#       REFERENCES hourlyweather (year, month, day, hour) MATCH SIMPLE
#       ON UPDATE NO ACTION ON DELETE NO ACTION;
#
# -- Foreign Key: startweekend_fkey
#
# -- ALTER TABLE trips DROP CONSTRAINT startweekend_fkey;
#
# ALTER TABLE trips
#   ADD CONSTRAINT startweekend_fkey FOREIGN KEY (startyear, startmonth, startday, starthour)
#       REFERENCES weekends (year, month, day, hour) MATCH SIMPLE
#       ON UPDATE NO ACTION ON DELETE NO ACTION;
```

To perform my hourly analysis, I aggregated the trips table by hour, and then joined this with the weather and weekend data to create a summary table:

```
# CREATE TABLE hourlyrides AS
#
# SELECT
#   startyear AS year,
#   startmonth AS month,
#   startday AS day,
#   starthour AS hour,
#   count(*)
# FROM
#   trips
# GROUP BY
#   startyear,
#   startmonth,
#   startday,
#   starthour;
#
# CREATE TABLE hourlyrideswithweather AS
#
# SELECT
#   hourlyrides.year,
#   hourlyrides.month,
#   hourlyrides.day,
#   hourlyrides.hour,
#   count AS ridescount
#   hourlyweather.temp,
```

```
#   hourlyweather.dewpoint,
#   hourlyweather.humidity,
#   hourlyweather.pressure,
#   hourlyweather.visibility,
#   hourlyweather.windspeed,
#   hourlyweather.conditions,
#   weekends.weekend
#
# FROM
#   hourlyrides
# INNER JOIN
#   hourlyweather on hourlyweather.year = hourlyrides.year
#   AND hourlyweather.month = hourlyrides.month
#   AND hourlyweather.day = hourlyrides.day
#   AND hourlyweather.hour = hourlyrides.hour
# INNER JOIN
#   weekends ON weekends.year = hourlyrides.year
#   AND weekends.month = hourlyrides.month
#   AND weekends.day = hourlyrides.day
#   AND weekends.hour = hourlyrides.hour;
```

Now the data is ready for a regression model.

```
con <- dbConnect(RPostgreSQL::PostgreSQL(), user="postgres", password="sinaiA9xpsql",
                 dbname="bikeshare")

con
```

```
## <PostgreSQLConnection:(61068,0)>
```

```
fulldata <- dbReadTable(con, "hourlyrideswithweather")
```

Lets look at a few scatter plots of the data.

There are a lot of variables, but for explanatory purposes lets look at the scatter plots of the numeric data, and use colour and facet to display the categorical variables (focusing on hour and weather conditions.).

There were too many weather conditions to accurately show on a graph, so I created a "simplified weather conditions" variable, just divided into "Good Weather" and "Bad Weather", the definitions of which are below in the code:

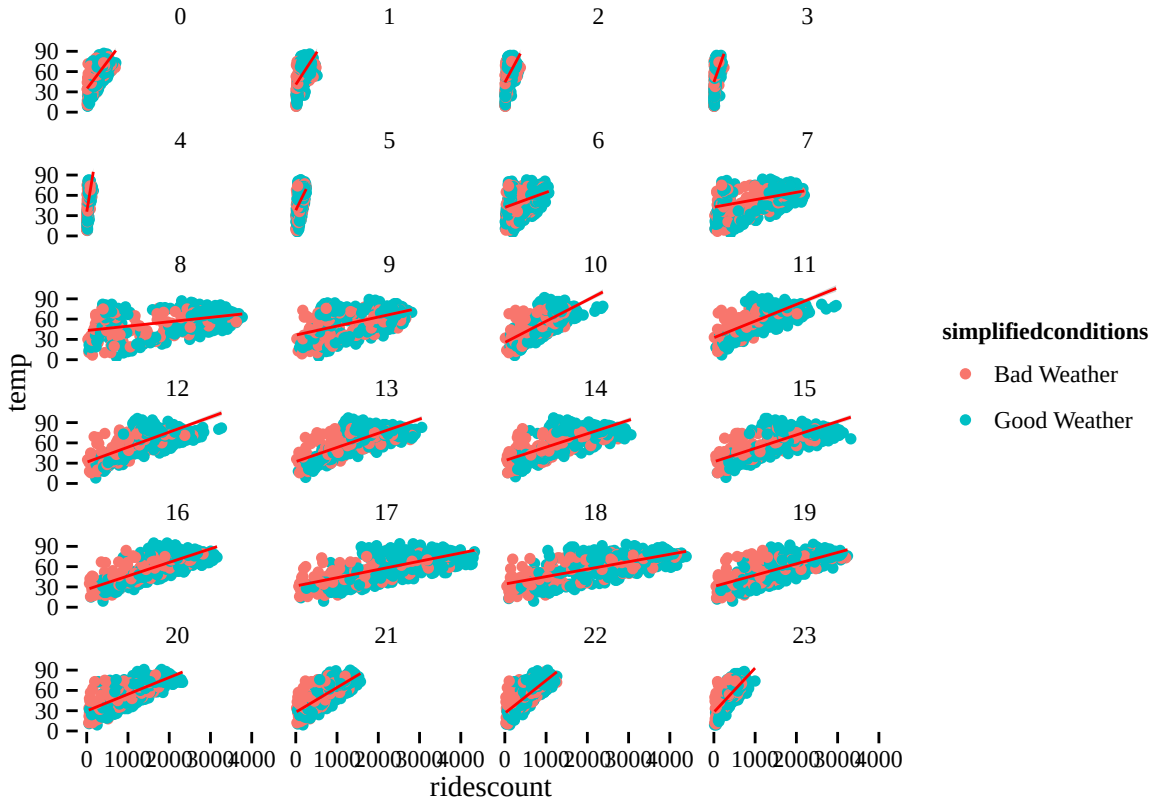
```
goodweather <- c("Clear", "Fog", "Haze", "Mostly Cloudy", "Overcase", "Partly Cloudy", "Scattered Clouds
badweather <- c("Heavy Rain", "Heavy Snow", "Light Freezing Rain", "Light Rain", "Light Snow",
                "Rain", "Snow", "Unknown")

fulldata$simplifiedconditions <- ifelse(fulldata$conditions %in% goodweather, "Good Weather", "Bad Weatl
```

And now we can look at a few graphs.

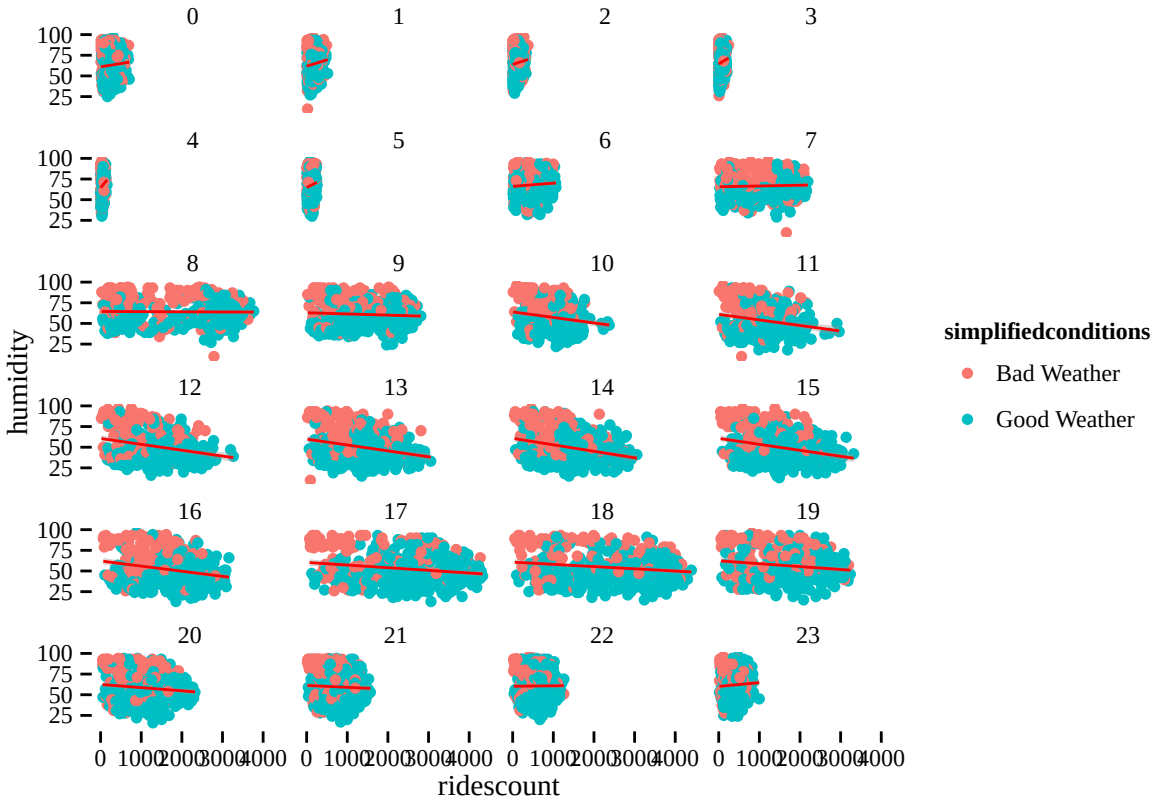
Temperature:

```
ggplot(fulldata, aes(x=ridescount, y=temp, colour = simplifiedconditions)) +
  geom_point() +
  geom_smooth(method="lm", color="red") +
  facet_wrap( ~ hour, nrow=6, ncol=4) +
  theme_tufte()
```



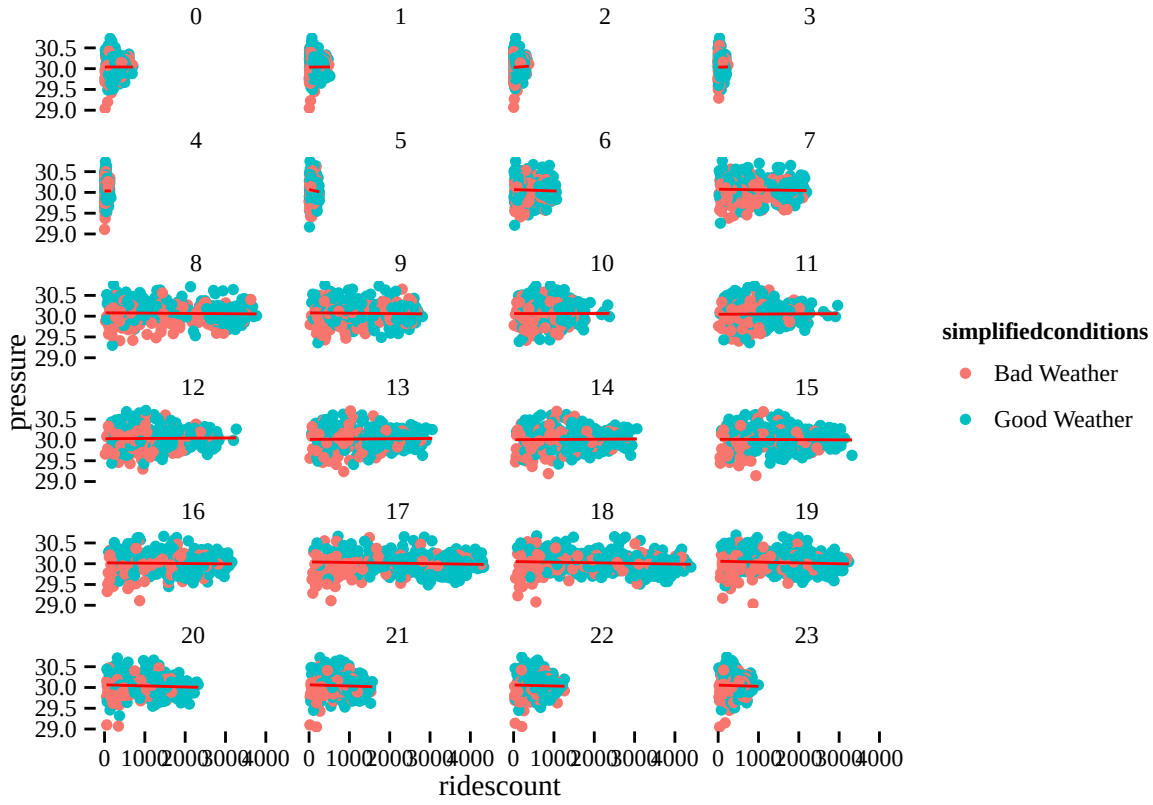
Humidity:

```
ggplot(fulldata, aes(x=ridescount, y=humidity, colour = simplifiedconditions)) +
  geom_point() +
  geom_smooth(method="lm", color="red") +
  facet_wrap( ~ hour, nrow=6, ncol=4) +
  theme_tufte()
```



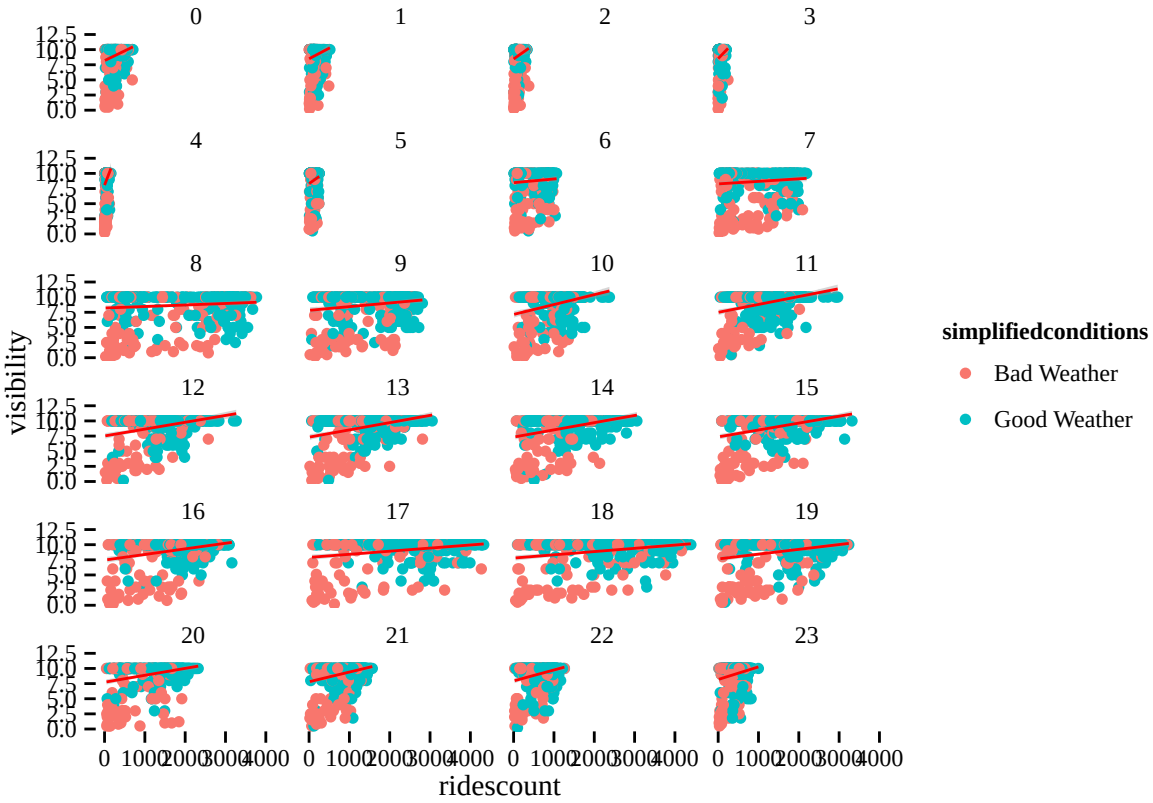
Pressure:

```
ggplot(fulldata, aes(x=ridescount, y=pressure, colour = simplifiedconditions)) +
  geom_point() +
  geom_smooth(method="lm", color="red") +
  facet_wrap( ~ hour, nrow=6, ncol=4) +
  theme_tufte()
```



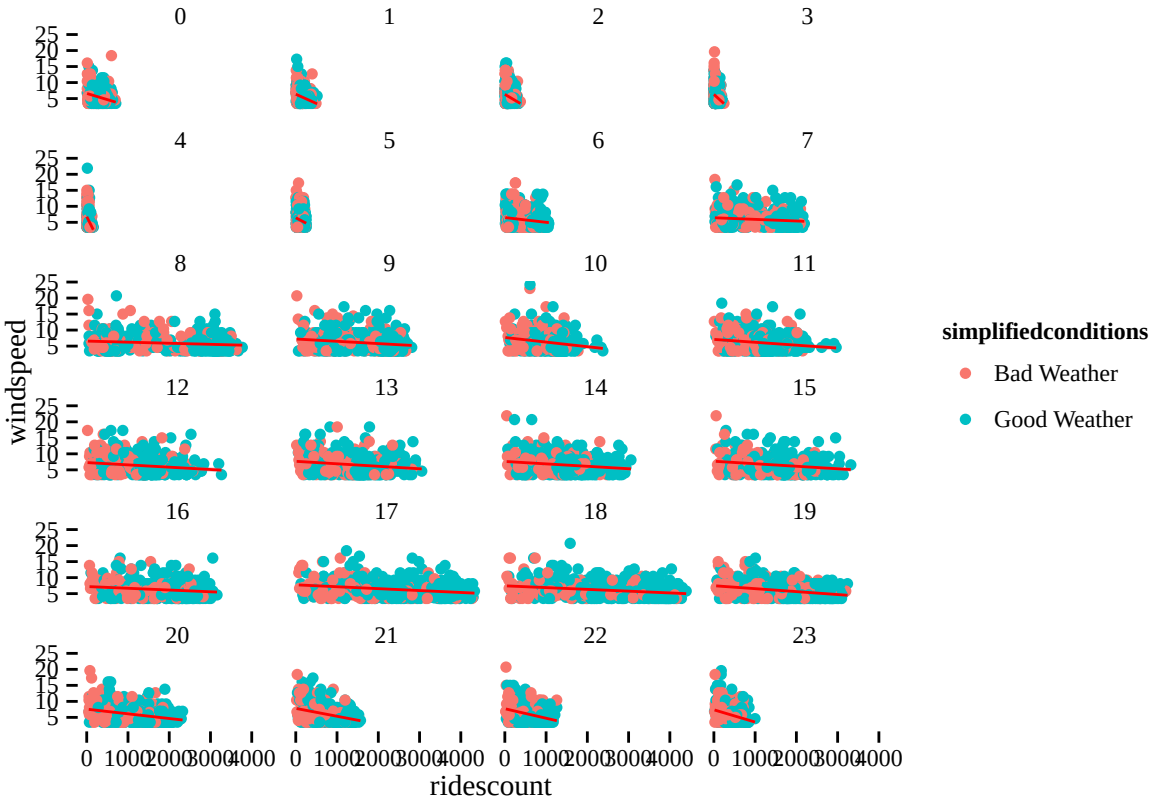
Visibility:

```
ggplot(fulldata, aes(x=ridescount, y=visibility, colour = simplifiedconditions)) +
  geom_point() +
  geom_smooth(method="lm", color="red") +
  facet_wrap( ~ hour, nrow=6, ncol=4) +
  theme_tufte()
```



Windspeed:

```
ggplot(fulldata, aes(x=ridescount, y=windspeed, colour = simplifiedconditions)) +
  geom_point() +
  geom_smooth(method="lm", color="red") +
  facet_wrap( ~ hour, nrow=6, ncol=4) +
  theme_tufte()
```


0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
windspeed
ridescount
simplifiedconditions
Bad Weather
Good Weather