💣

Linear Regression report

Get the Data

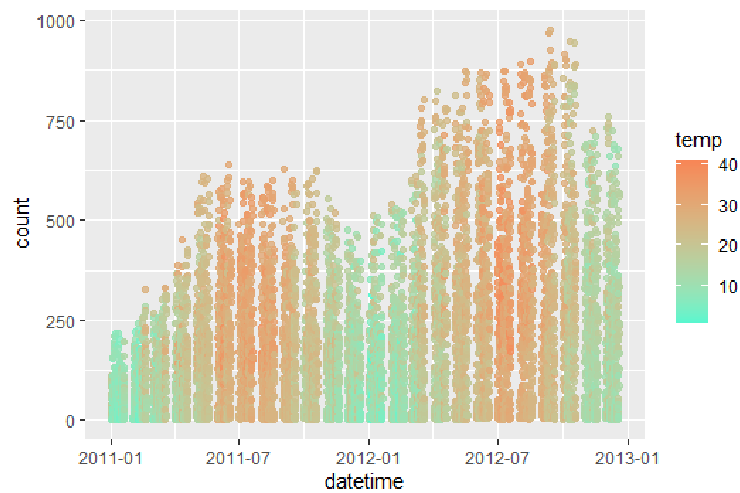
- datetime - hourly date + timestamp
- season - 1 = spring, 2 = summer, 3 = fall, 4 = winter
- holiday - whether the day is considered a holiday
- workingday - whether the day is neither a weekend nor holiday
- weather -
  - 1: Clear, Few clouds, Partly cloudy, Partly cloudy
  - 2: Mist + Cloudy, Mist + 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

```
# Read in bikeshare.csv file and set it to a dataframe called bike
# Check the head of df
bike <- read.csv("bikeshare.csv")
class(bike)
head(bike)
# Count is what we are trying to predict

# Exploratory Data Analysis
pl <- ggplot(df, aes(x = temp, y = count, color = temp))
pl + geom_point(alpha = 0.2)

# convert datetime column to POSIXct class
df$datetime <- as.POSIXct(df$datetime)
class(df$datetime)

pl <- ggplot(df, aes(x = datetime, y = count, colour = temp))
pl + geom_point(alpha = 0.8) + scale_colour_gradient(high= "#f78656", low = "#56f7cf")
```

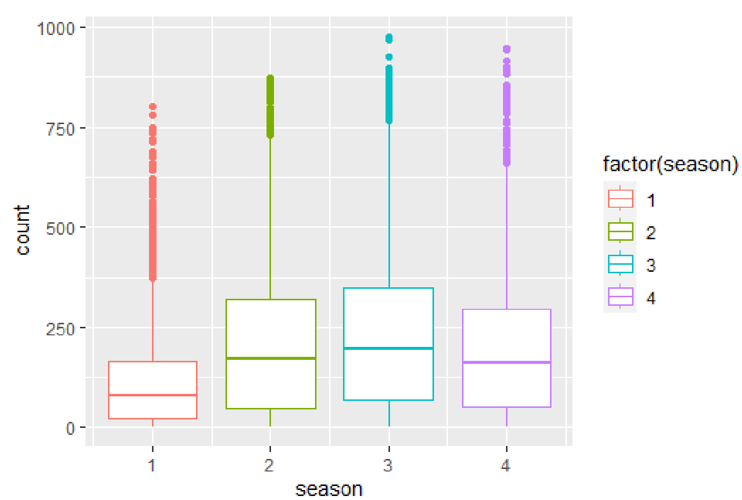



```
num.cols <- df[c("temp", "count")]
cor.data <- cor(num.cols)

#sol
cor(bike[,c('temp','count')])
```

→ **explore the season data. Create a boxplot, with the y axis indicating count and the x axis begin a box for each season.**

```
pl2 <- ggplot(df, aes(x = season, y = count, colour = factor(season)))
pl2 + geom_boxplot()
```



we found:

- we found A line can't capture a non-linear relationship
- มีการเช่าในฤดูหนาวมากกว่าในฤดูใบไม้ผลิ

Feature Engineering

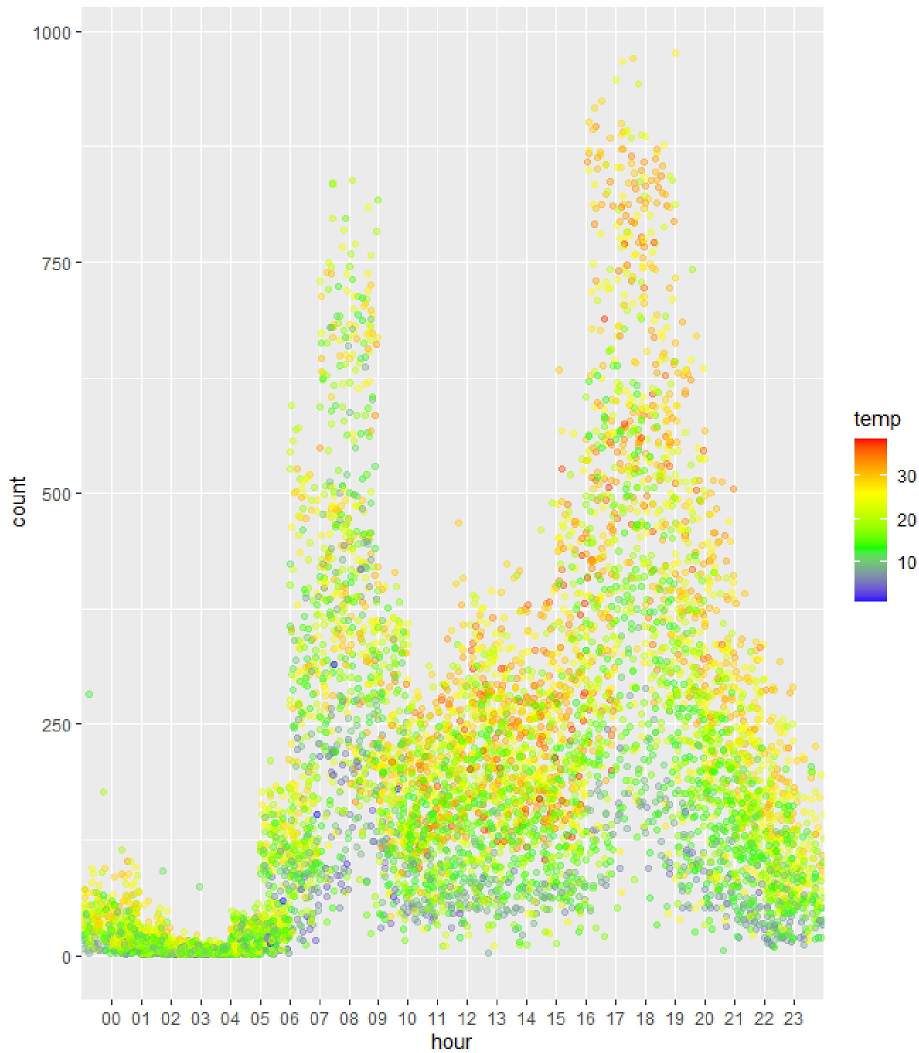
→ **Create an "hour" column that takes the hour from the datetime column.**

```
hr_time <- df[,1]
df$hour <- format(hr_time, "%H")
# or
format(df$datetime, "%H")

# sol
bike$hour <- sapply(bike$datetime,function(x){format(x,"%H")})
```

- **workingday == 1**

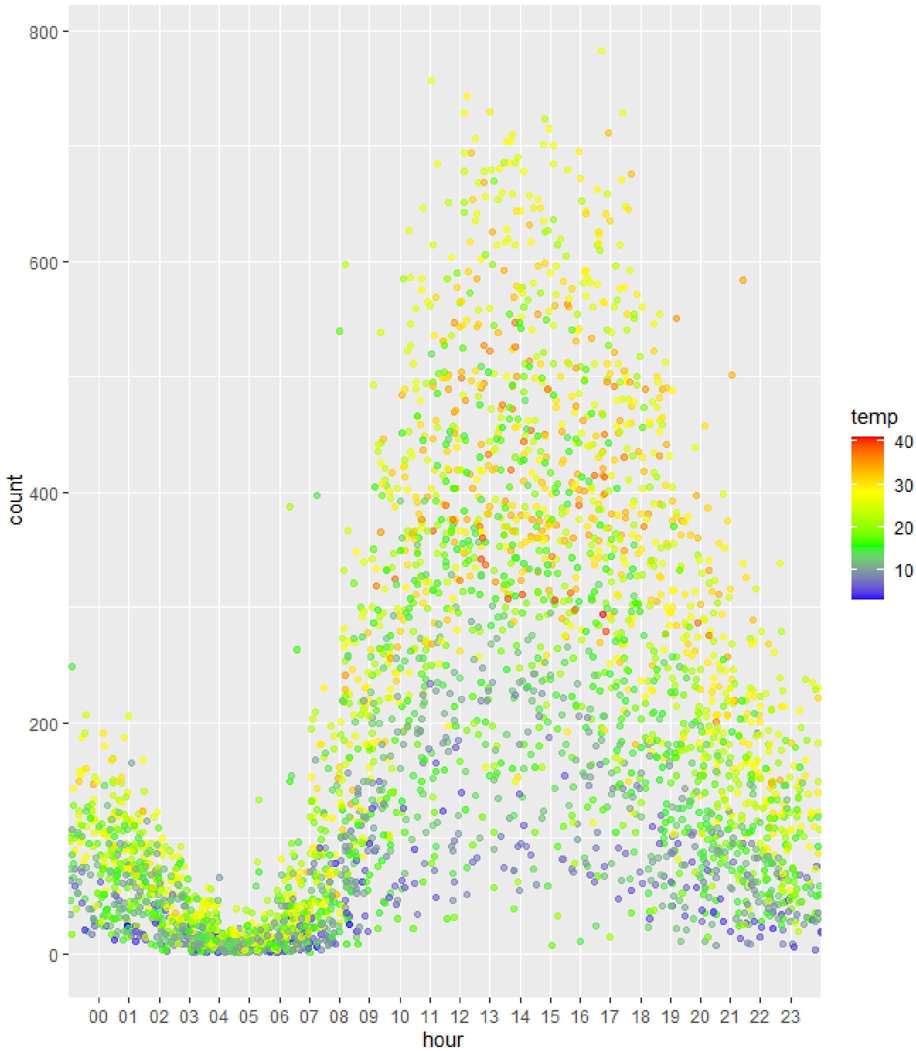
```
# workingday
df_w <- df[df$workingday == 1,]
pl3 <- ggplot(df_w, aes(y = count, x = hour, colour = temp))
pl3 + geom_point(position=position_jitter(w=1, h=0), alpha = 0.4) + scale_colour_gradientn(colors=c('blue','green','yellow','red'))
```



- **non workingdays == 0**

```
# non workingdays
df_nw <- df[df$workingday == 0,]
```

```
pl4 <- ggplot(df_nw, aes(y = count, x = hour, colour = temp))
pl4 + geom_point(position=position_jitter(w=1, h=0), alpha = 0.6) + scale_colour_gradientn(colors=c('blue','green','yellow','red'))
```



- we found peak activity during the morning (~8am) and right after work gets out (~5pm)

Building the Model

→ Use lm() to build a model that predicts count based solely on the temp feature, name it temp.mode

```
# Building the Model
# count(temp)
temp.model <- lm(count ~ temp,data = df)
summary(temp.model)
```

Interpreting the intercept (β0):

- It is the value of y when x=0.
- Thus, it is the estimated number of rentals when the temperature is 0 degrees Celsius.
- Note: It does not always make sense to interpret the intercept

Interpreting the "temp" coefficient (β1):

- It is the change in y divided by change in x, or the "slope".
- Thus, a temperature increase of 1 degree Celsius is associated with a rental increase of 9.17 bikes.
- This is not a statement of causation.
- β1 would be negative if an increase in temperature was associated with a decrease in rentals.

Test an temp 25

- Using the values we just got above
- Using the predict() function

```
# y = b0 + b1*x
temp.model <- lm(count ~ temp,data = df)
summary(temp.model)
# b0 = (Intercept) = 6.0462 , b1 = temp = 9.1705
y = 6.0462 + 9.17*25
y = 235.2962

temp.test <- data.frame(temp=c(25))
p <- predict(temp.model,temp.test)
p = 235.3097

# change the hour column to a column of numeric values
df$hour <- sapply(df$hour,as.numeric)
```

count → prediction with

- season
- holiday
- workingday
- weather
- temp
- humidity
- windspeed
- hour (factor)

```
# split data
set.seed(33)
sample <- sample.split(df$temp, SplitRatio = 0.70)
train = subset(df, sample == TRUE)
test = subset(df, sample == FALSE)

# Built model
model <- lm(count ~ . -casual - registered -datetime -atemp,df = train)
summary(model)

# or
model <- lm(count ~ season  + holiday + workingday + weather + temp + humidity + windspeed + hour,data = train)

# predict data
p <- predict(model,test)
mean(p)
results <- cbind(p,test$count)
colnames(results) <- c('pred','real')
results <- as.data.frame(results)

# mean squared error
```

```
mse <- mean((results$real-results$pred)^2)

# root mean squared error
mse^0.5

# R-Squared Value
SSE <- sum((results$pred - results$real)^2)
SST <- sum( (mean(df$count) - results$real)^2)
R2 <- 1 - SSE/SST
```

- we found model doesn't work well given our seasonal and time series data