UBER Question 4 Notebook

Brad Chattergoon

Contents

```r
library(tidyverse)
library(magrittr)
library(jsonlite)
library(lubridate)
library(grid)
library(gridExtra)

knitr::opts_chunk$set(echo = TRUE)
```

```r
logins = fromJSON("logins.json") %>%
  tibble() %>%
  set_colnames(c("login_timestamp")) %>%
  mutate(login_timestamp = ymd_hms(login_timestamp)) %>%
  mutate(
    year = year(login_timestamp),
    month = month(login_timestamp),
    week = week(login_timestamp),
    day = day(login_timestamp),
    weekday = wday(login_timestamp,
                   label = TRUE,
                   abbr = TRUE,
                   week_start = getOption("lubridate.week.start",7)),
    hour = hour(login_timestamp)
  )

summary(logins)
```

```
##  login_timestamp                         year          month            week
##  Min.   :2012-03-01 00:05:55   Min.   :2012   Min.   :3.000   Min.   : 9.00
##  1st Qu.:2012-03-18 04:51:39   1st Qu.:2012   1st Qu.:3.000   1st Qu.:12.00
##  Median :2012-04-04 02:02:30   Median :2012   Median :4.000   Median :14.00
##  Mean   :2012-04-02 23:22:04   Mean   :2012   Mean   :3.549   Mean   :13.77
##  3rd Qu.:2012-04-19 15:29:40   3rd Qu.:2012   3rd Qu.:4.000   3rd Qu.:16.00
##  Max.   :2012-04-30 23:59:29   Max.   :2012   Max.   :4.000   Max.   :18.00
##
##       day         weekday          hour
##  Min.   : 1.00   Sun:5173   Min.   : 0.00
##  1st Qu.: 9.00   Mon:2139   1st Qu.: 3.00
##  Median :17.00   Tue:1861   Median :11.00
##  Mean   :16.49   Wed:2155   Mean   :10.84
##  3rd Qu.:24.00   Thu:2857   3rd Qu.:19.00
##  Max.   :31.00   Fri:3198   Max.   :23.00
##                  Sat:5064
```

We see that the data is all for 2012 and covers the period from March 1st through April 30th, 2 months of data. We also see that there are a lot more logins on the weekends than during the week.

```
logins_plot = logins %>%
  ggplot(aes(x = login_timestamp)) +
  geom_histogram() +
  labs(title = "Histogram of Login Data") +
  xlab("Login Timestamp")

hourly = logins %>%
  group_by(year, month, day, hour) %>%
  summarise(num_logins = n()) %>%
  ungroup() %>%
  mutate(timestamp = ymd_h(paste(year, month, day, hour, sep = "-")))
```

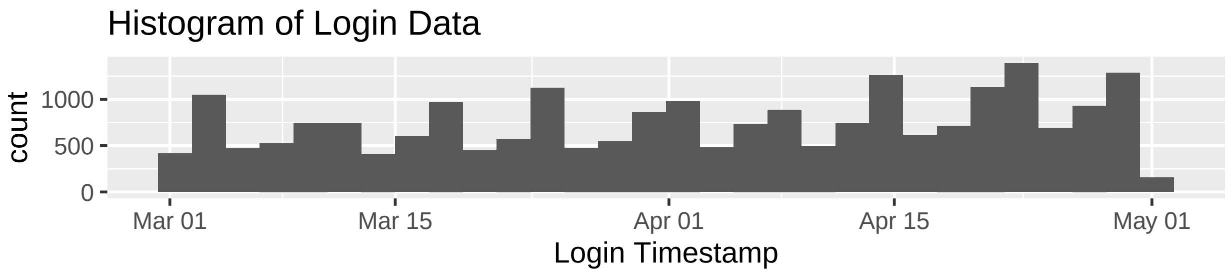
```
## `summarise()` regrouping output by 'year', 'month', 'day' (override with `.groups` argument)
```

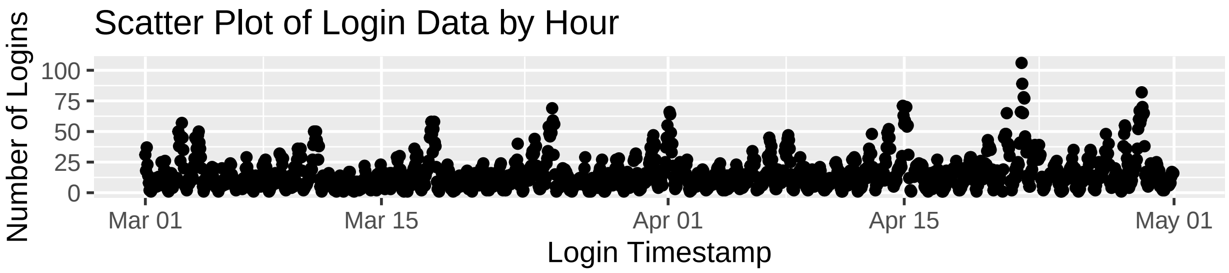
```
hourly_plot = hourly %>%
  ggplot(aes(x = timestamp, y = num_logins)) +
  geom_point() +
  labs(title = "Scatter Plot of Login Data by Hour") +
  xlab("Login Timestamp") +
  ylab("Number of Logins")

daily = logins %>%
  group_by(year, month, day) %>%
  summarise(num_logins = n()) %>%
  ungroup() %>%
  mutate(timestamp = ymd(paste(year, month, day, sep = "-")))
```

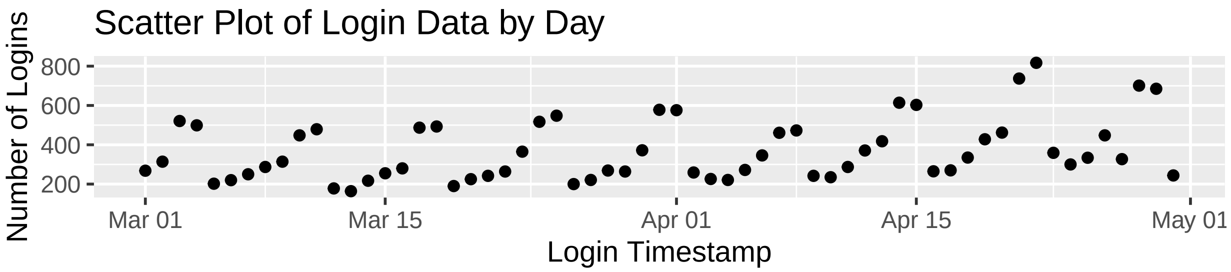
```
## `summarise()` regrouping output by 'year', 'month' (override with `.groups` argument)
```

```
daily_plot = daily %>%
  ggplot(aes(x = timestamp, y = num_logins)) +
  geom_point() +
  labs(title = "Scatter Plot of Login Data by Day") +
  xlab("Login Timestamp") +
  ylab("Number of Logins")

grid.arrange(logins_plot, hourly_plot, daily_plot,nrow = 3)
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```







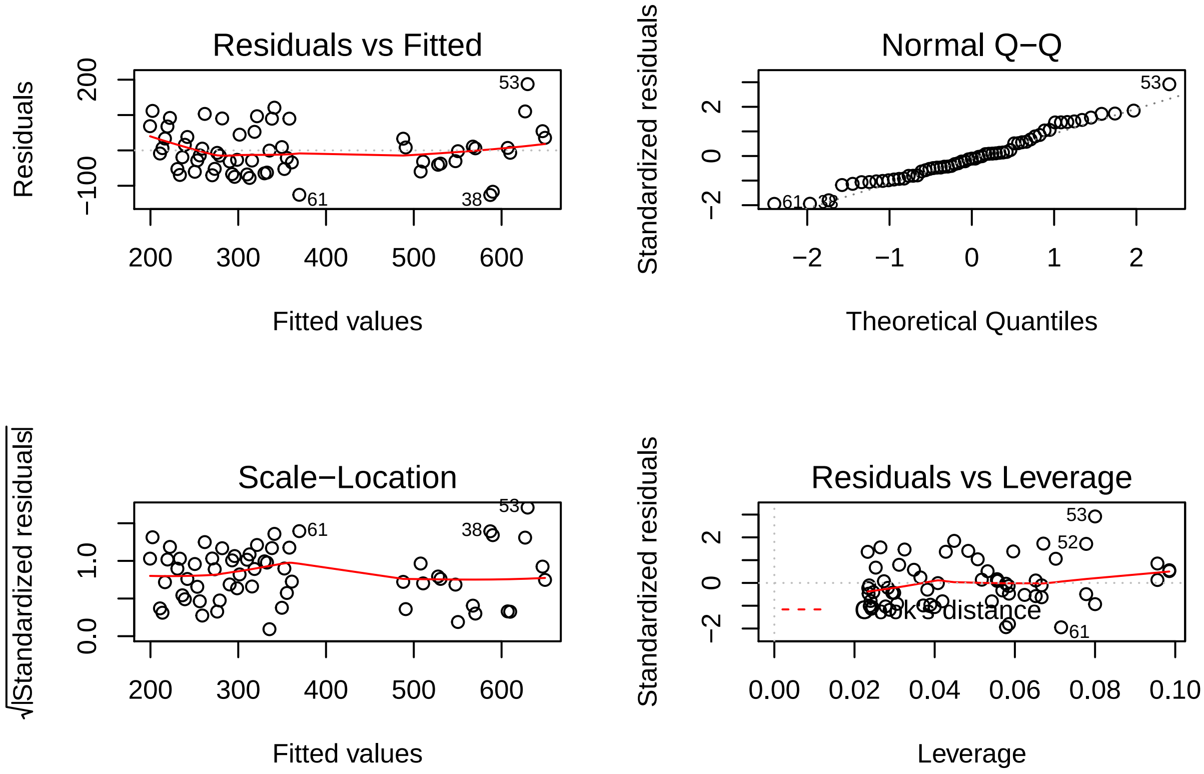
We see in the plots that there is a cyclical trend in the data. We see this most clearly in the plot of the data aggregated by day. The period of the cycle seems to be 1 week. In modeling the data we will need to account for this behavior. In order to identify the long-term trends in this data we will use the day level of aggregation and estimate how number of logins change by day.

We first try including an indicator variable for weekend in an effort to parse out different intercepts for weekend vs non-weekend, assuming similar underlying behavior.

```
daily = daily %>%
  mutate(day_number = c(1:nrow(daily))) %>%
  mutate(weekend = ifelse(wday(timestamp, label = TRUE, abbr = TRUE) %in% c("Sun","Sat"), 1, 0))

reg_1 = lm(num_logins ~ day_number + weekend, data = daily)
summary(reg_1)
```

```
##
## Call:
## lm(formula = num_logins ~ day_number + weekend, data = daily)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -126.145  -52.210   -7.044   37.967  187.342
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 196.6858    18.1385  10.844 1.42e-15 ***
## day_number    2.8342     0.4869   5.821 2.69e-07 ***
## weekend     282.7584    18.7973  15.043  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 66.95 on 58 degrees of freedom
## Multiple R-squared:  0.8196, Adjusted R-squared:  0.8134
## F-statistic: 131.7 on 2 and 58 DF,  p-value: < 2.2e-16
```

```
old.par <- par(mfrow=c(2,2))
plot(reg_1)
```



```
par(old.par)
```

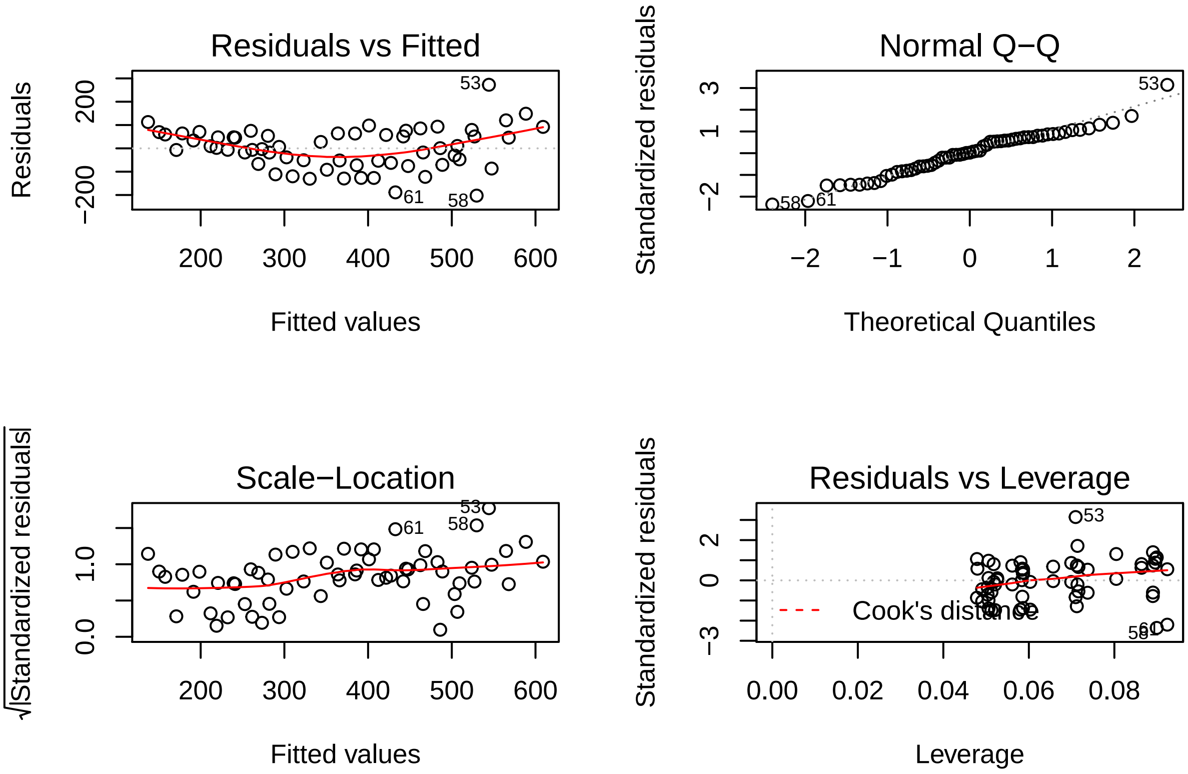
We find that the model with an indicator variable for weekend performs fairly well. We get an adjusted R2 of 0.8134 and the diagnostic plots show fairly good fit with fairly straight lines along 0 for the residuals plots and the scale-location plot doesn't show any significant strange behavior. The normal Q-Q plot also seems fairly normal.

Noting the clear cyclical behavior, we also endeavor to test a model with cyclical elements. We will use sin and cos terms in the regression to capture the cyclic behavior.

```
daily = daily %>%
  mutate(
    sint = sin(2*pi*day_number/7),
    cost = cos(2*pi*day_number/7)
    )

reg_2 = lm(num_logins ~ day_number + sint + cost, data = daily)
summary(reg_2)
```

```
##
## Call:
## lm(formula = num_logins ~ day_number + sint + cost, data = daily)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -202.665  -62.342    2.025   63.596  272.664
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  272.5104    23.3608  11.665  < 2e-16 ***
## day_number     2.9235     0.6554   4.461 3.90e-05 ***
## sint          54.2688    16.2157   3.347  0.00145 **
## cost        -155.8593    16.4146  -9.495 2.42e-13 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 89.97 on 57 degrees of freedom
## Multiple R-squared:  0.6798, Adjusted R-squared:  0.6629
## F-statistic: 40.33 on 3 and 57 DF,  p-value: 4.079e-14
```

```
old.par <- par(mfrow=c(2,2))
plot(reg_2)
```



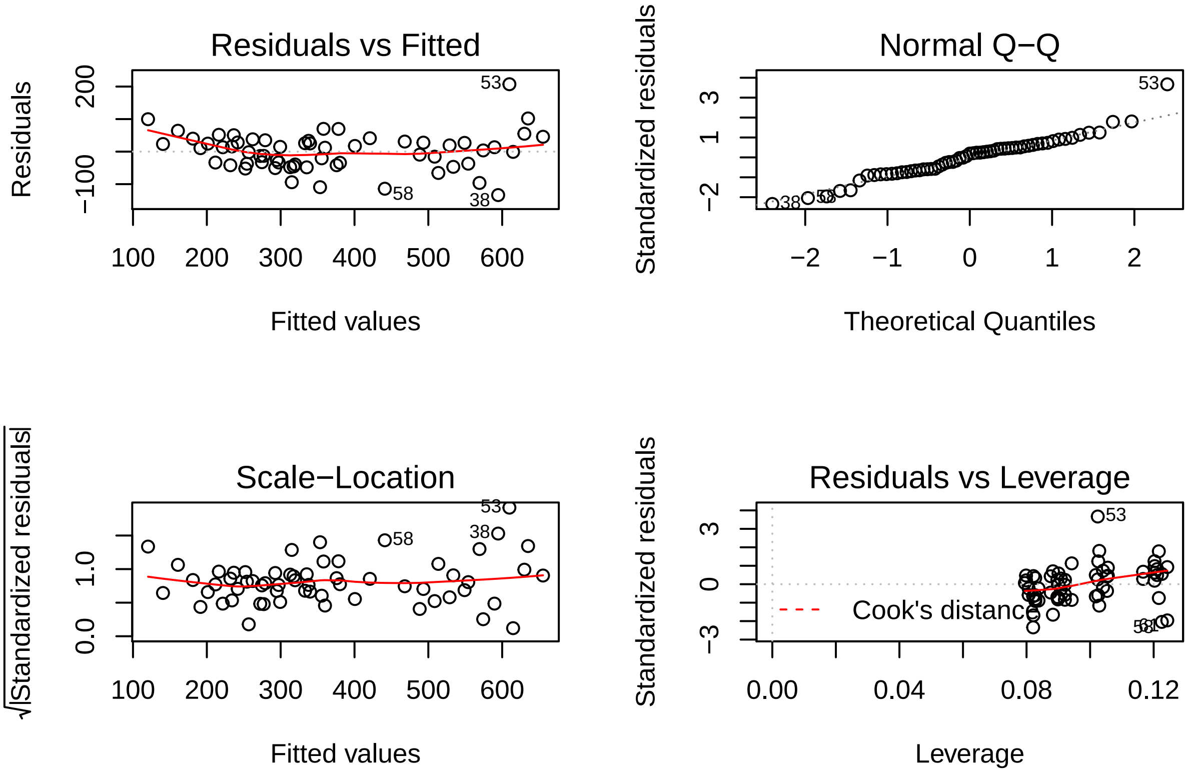
```
par(old.par)
```

We try a regression with one set of cycle terms, i.e. sint and cost. We see that the residuals plot shows some remaining unaccounted for cyclic behavior so we add in another set of cyclical terms to account for this.

```r
daily = daily %>%
  mutate(
    sin2t = sin(4*pi*day_number/7),
    cos2t = cos(4*pi*day_number/7)
    )

reg_3 = lm(num_logins ~ day_number + sint + cost + sin2t + cos2t, data = daily)
summary(reg_3)
```

```
##
## Call:
## lm(formula = num_logins ~ day_number + sint + cost + sin2t +
##     cos2t, data = daily)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -133.57  -37.31   10.64   30.94  207.19
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  274.4973    15.4807  17.732  < 2e-16 ***
## day_number     2.8929     0.4343   6.661 1.35e-08 ***
## sint          54.9946    10.7478   5.117 4.08e-06 ***
## cost        -153.4351    10.8830 -14.099  < 2e-16 ***
## sin2t         12.5129    10.7940   1.159    0.251
## cos2t         92.7442    10.8095   8.580 1.00e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 59.61 on 55 degrees of freedom
## Multiple R-squared:  0.8644, Adjusted R-squared:  0.852
## F-statistic: 70.09 on 5 and 55 DF,  p-value: < 2.2e-16
```

```r
old.par <- par(mfrow=c(2,2))
plot(reg_3)
```



```r
par(old.par)

confint(reg_3)
```

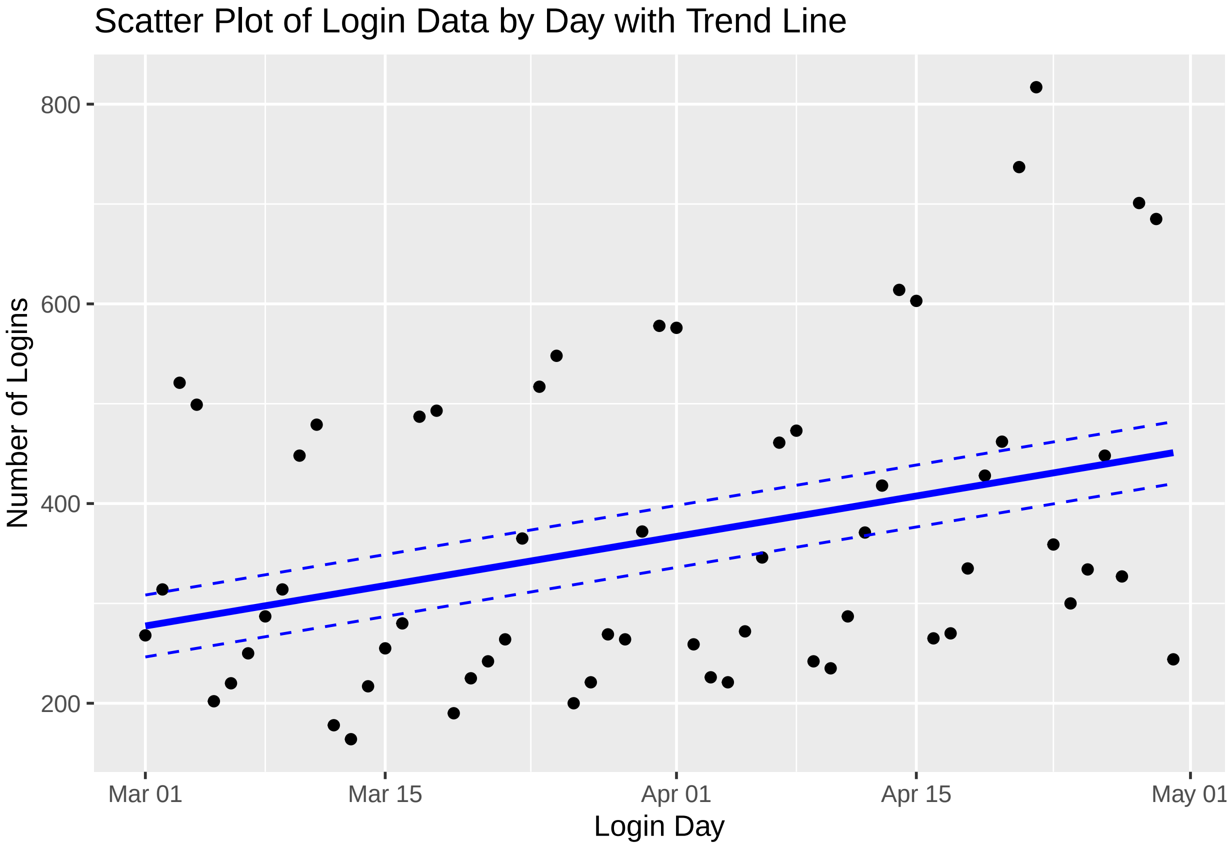
```
##                   2.5 %      97.5 %
## (Intercept)  243.473254  305.521358
## day_number     2.022588    3.763213
## sint          33.455580   76.533525
## cost        -175.245237 -131.625056
## sin2t         -9.118841   34.144646
## cos2t         71.081468  114.406944
```

With the inclusion of the additional level of cyclic terms we see that the adjusted R2 improves significantly, up to a respectable 0.852, and we see better behavior in the residuals plot in the form of a mostly straight line. There is some odd behavior for the first few observations but overall the residuals plot is indicating good fit. The normal Q-Q plot shows good normality behavior in the model as well, with the exception of obs 53 which may be indicating some outlier behavior but I am unable to describe this further so I leave it in the model. The sin2t variable is shown to be insignificant in the model, but significance on this variable alternates with its corresponding cosine variable depending on the start day label (i.e. 1 vs 0), so this is likely capturing whether we start at a 0 or 1. Given that in this exercise we are focused on the long-term trend and only controlling for the cyclical behavior this should not be of great concern in the modeling.

We find a very similar long-term trend in both the cyclical and variable intercept models, but we find a better adjusted R2 using the cyclical modeling compared to the variable intercept modeling so we accept the former model as the "correct" one. With additional data we can attempt to test the predictive power of the model but given the limited size of the dataset we avoid this in this exercise.

In conclusion, based on the model estimated, the long-term trend seems to be an increase in approximately 2.8929 logins per day over the period of the sample. We include this regression line on the daily login plot with confidence intervals for the intercept.

```
daily_plot = daily %>%
  mutate(
    reg_data = 274.4973 + 2.8929 * day_number,
    reg_data_lower = 243.473254 + 2.8929 * day_number,
    reg_data_upper = 305.521358 + 2.8929 * day_number
    ) %>%
  ggplot(aes(x = timestamp, y = num_logins)) +
  geom_point() +
  labs(title = "Scatter Plot of Login Data by Day with Trend Line") +
  xlab("Login Day") +
  ylab("Number of Logins") +
  geom_line(aes(y = reg_data), col = "blue", size = 1.2) +
  geom_line(aes(y = reg_data_lower), col = "blue", linetype = "dashed") +
  geom_line(aes(y = reg_data_upper), col = "blue", linetype = "dashed")

daily_plot
```



We now explore the data in the context of day of week logins and hourly logins.

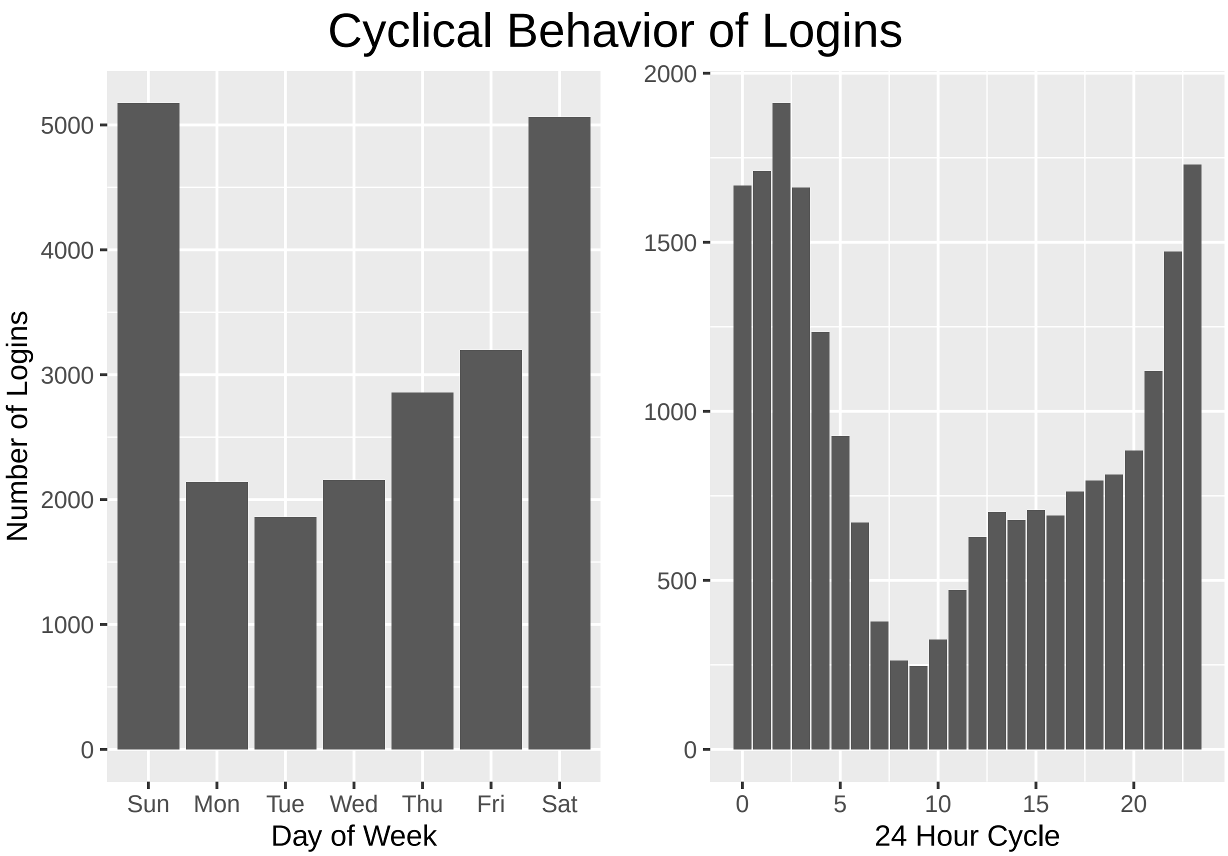
```
week_day = logins %>%
  group_by(weekday) %>%
  summarise(num_logins = n())
```

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

```
week_day_plot = week_day %>%
  ggplot(aes(x = weekday, y = num_logins)) +
  geom_bar(stat = "identity") +
  ylab("Number of Logins") +
  xlab("Day of Week")

hourly = logins %>%
  group_by(hour) %>%
  summarise(num_logins = n())
```

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

```
hourly_plot = hourly %>%
  ggplot(aes(x = hour, y = num_logins)) +
  geom_bar(stat = "identity") +
  ylab(element_blank()) +
  xlab("24 Hour Cycle")

grid.arrange(week_day_plot, hourly_plot, ncol = 2, top = textGrob("Cyclical Behavior of Logins", gp=gpa
```



We start with discussing the weekday cyclical trends. We see that the day with the lowest number of logins is Tuesday while the highest logins are on the weekend, with Sunday slightly higher than Saturday. From the low on Tuesday we see a steady increase in login activity peaking on Sunday and sharply falling back down on Monday. When we compare this with the daily plots of logins, we note that Mon, Tue, and Wed are often variable in the which day has the largest logins, and the difference between Tue and Mon and Tue and

Wednesday is unlikely to be significant. We can consider these days very similar from a login perspective and, assuming login behavior is representative of demand, we can consider these the same from a demand perspective.

With the above in mind, my hypothesis for the weekday trend is that it follows social/non-work behavior, with the idea being that Uber demand is higher for transportation when used for these types of activities. As is the conventional wisdom “no one goes to the club on Tuesday”, so we see Mon-Wed with lower demand and then demand ticks upward after “hump-day” (Wednesday) with the soft-start of the weekend on Thursday. When the weekend is in full swing on Saturday and Sunday, logins are more than double the average workday levels.

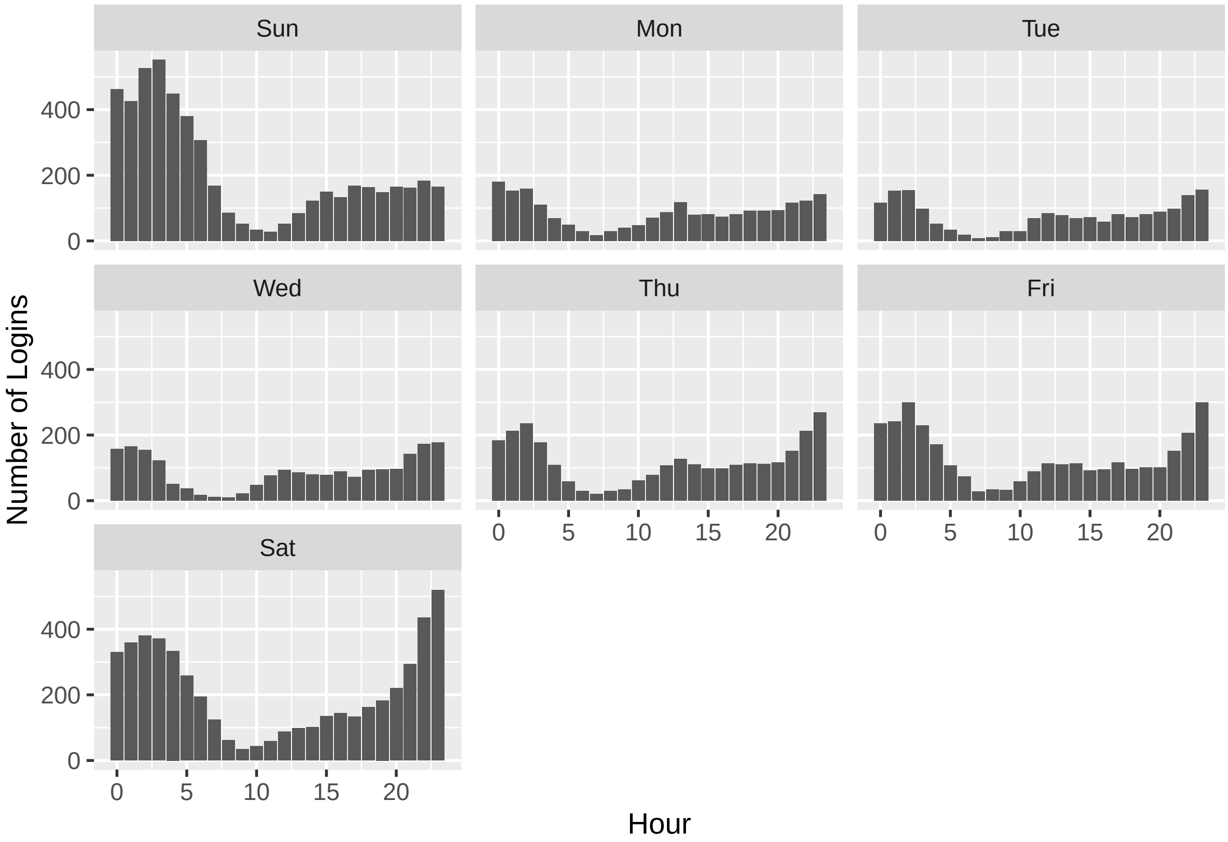
When we look at the 24 hour cycle we see a trend that fits a similar hypothesis as the one for the weekday cycle. Login activity is at a minimum at 9a, generally when most people in the Uber target market (those with disposable income in areas that car ownership is low such as a major city) are meant to be at work. It then starts ticking upward each hour until 2a at which point it begins ticking downward until 9a. My hypothesis for what’s happening here is that from around noon to 8p we see logins based on behavior such as going out for lunch and perhaps after work drinks or other activities. The large spikes after 8p are a combination of 2 things. The first one is people returning home late and perhaps not wanting to take public transportation during weekdays, or public transit might be offline at this time. We will think of this as baseline behavior. The second contributor is the large spike in logins on the weekends and the “soft” weekends (i.e. Thu, Fri). During weekends lots of people explore nightlife and return home late. The observation that logins peak at 2a and then trail off suggest that this is a city in a state with a 2a legal order for barclosures such as Boston since this is East Coast data. For reference, if this hypothesis is true then in a city like New York we should see the spiking behavior peak at 4a since this is when bars stop serving alcohol in that city. The baseline behavior plus the spiking behavior on weekends makes it such that the aggregated logins by hour are much larger at night time.

We can gather some evidence for this hypothesis by looking at the number of logins across hours by weekday.

```
weekday_hourly = logins %>%
  group_by(weekday, hour) %>%
  summarise(num_logins = n())
```

```
## ‘summarise()‘ regrouping output by ’weekday’ (override with ‘.groups‘ argument)
```

```
weekday_hour_plot = weekday_hourly %>%
  ggplot(aes(x = hour, y = num_logins)) +
  geom_bar(stat = "identity") +
  xlab("Hour") +
  ylab("Number of Logins") +
  facet_wrap(~ weekday)

weekday_hour_plot
```



When we look at the graphs we see some evidence suggesting that this split into "baseline" behavior from Mon, Tue, Wed, and somewhat Thu and Fri, and the "spiking" behavior on the weekends, and to a lesser extent the "soft" weekend, does seem to exist.