Linear Regression Project

We'll work on the Bike Sharing Demand Kaggle challenge! We won't submit any results to the competition, but feel free to explore Kaggle more in depth.

Instructions

Just complete the tasks outlined below.

Get the Data

You can download the data or just use the supplied csv in the repository. The data has the following features:

- 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.

In [7]:

```
bike <- read.csv('bikeshare.csv')
```

Check the head of df

In [8]:

```
head(bike)
```

Out[8]:

	datetime	season	holiday	workingday	weather	temp	atemp	humidity	windspeed	ca
1	2011-01-01 00:00:00	1	0	0	1	9.84	14.395	81	0	3
2	2011-01-01 01:00:00	1	0	0	1	9.02	13.635	80	0	8
3	2011-01-01 02:00:00	1	0	0	1	9.02	13.635	80	0	5
4	2011-01-01 03:00:00	1	0	0	1	9.84	14.395	75	0	3
5	2011-01-01 04:00:00	1	0	0	1	9.84	14.395	75	0	0
6	2011-01-01 05:00:00	1	0	0	2	9.84	12.88	75	6.0032	0

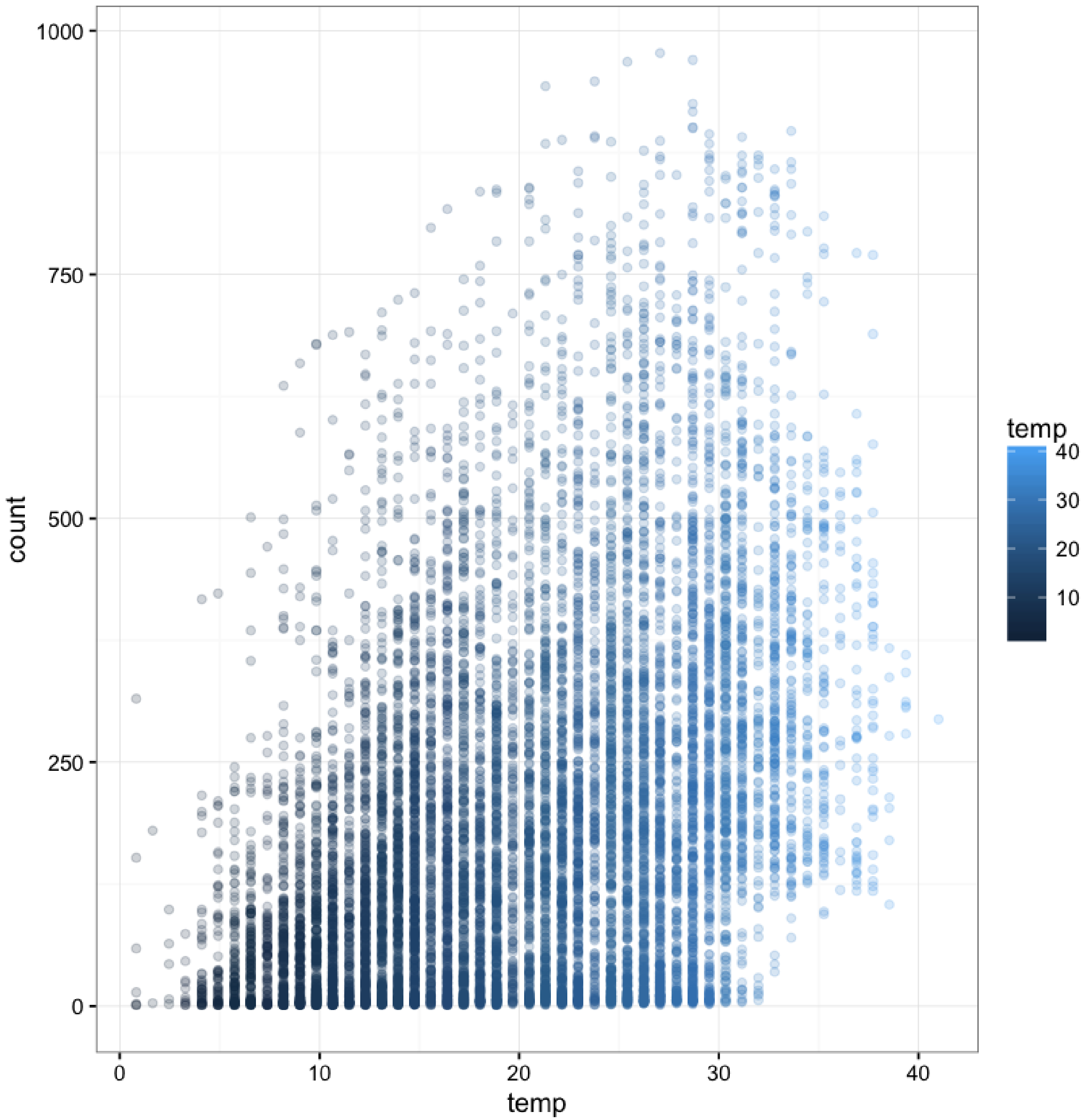
Can you figure out what is the target we are trying to predict? Check the Kaggle Link above if you are confused on this.

In [9]:
```
# Count is what we are trying to predict
```

Exploratory Data Analysis

Create a scatter plot of count vs temp. Set a good alpha value.

In [27]:
```
library(ggplot2)
ggplot(bike,aes(temp,count)) + geom_point(alpha=0.2, aes(color=temp)) +
 theme_bw()
```

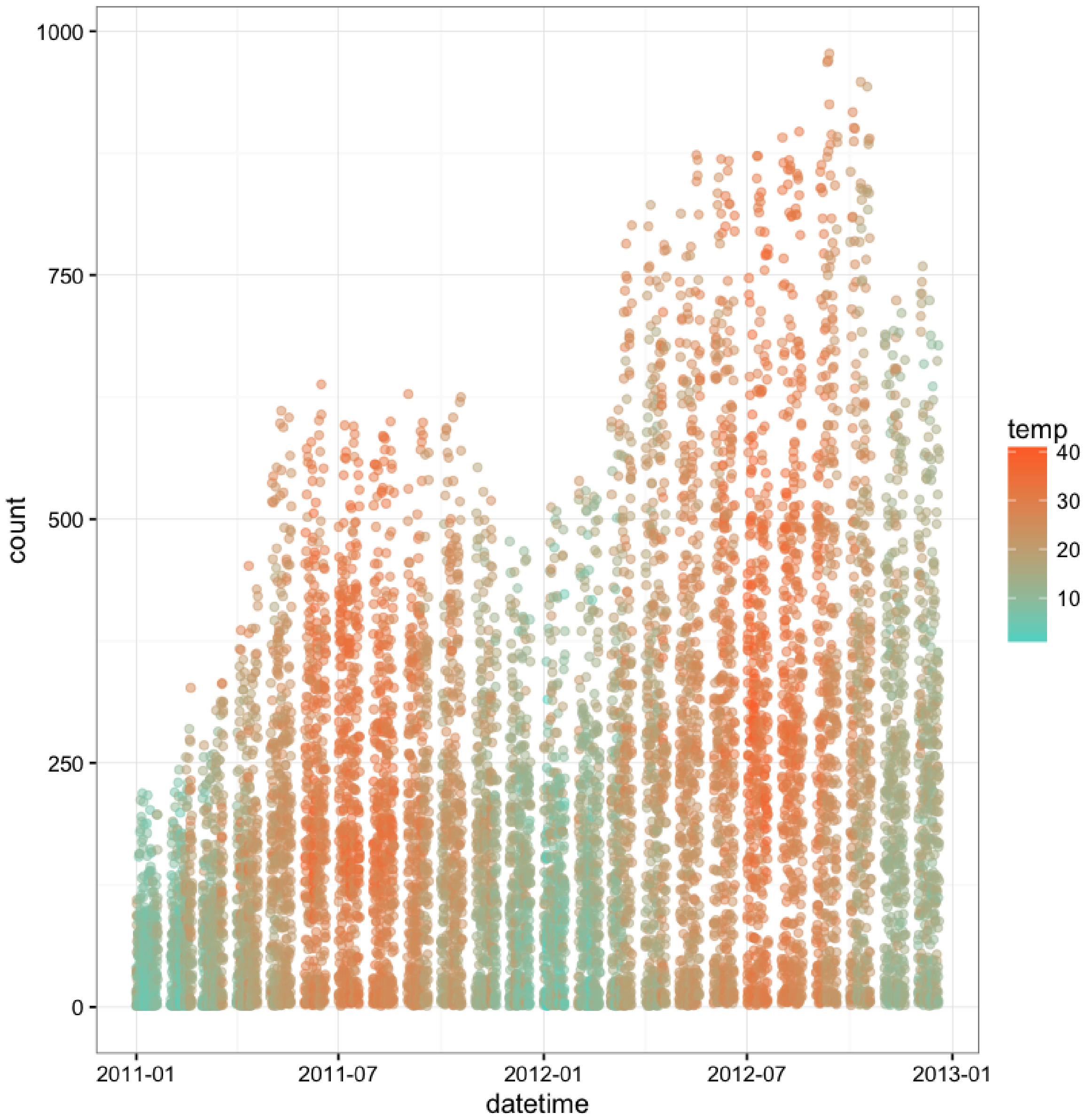


Plot count versus datetime as a scatterplot with a color gradient based on temperature.

You'll need to convert the datetime column into POSIXct before plotting.

In [12]:
```
bike$datetime <- as.POSIXct(bike$datetime)
```

In [26]:
```
ggplot(bike,aes(datetime,count)) + geom_point(aes(color=temp),alpha=0.5
)  + scale_color_continuous(low='#55D8CE',high='#FF6E2E') +theme_bw()
```



Hopefully you noticed two things: A seasonality to the data, for winter and summer. Also that bike rental counts are increasing in general. This may present a problem with using a

linear regression model if the data is non-linear. Let's have a quick overview of pros and cons right now of Linear Regression:

Pros:

- Simple to explain
- Highly interpretable
- Model training and prediction are fast
- No tuning is required (excluding regularization)
- Features don't need scaling
- Can perform well with a small number of observations
- Well-understood

Cons:

- Assumes a linear relationship between the features and the response
- Performance is (generally) not competitive with the best supervised learning methods due to high bias
- Can't automatically learn feature interactions

We'll keep this in mind as we continue on. Maybe when we learn more algorithms we can come back to this with some new tools, for now we'll stick to Linear Regression.

What is the correlation between temp and count?

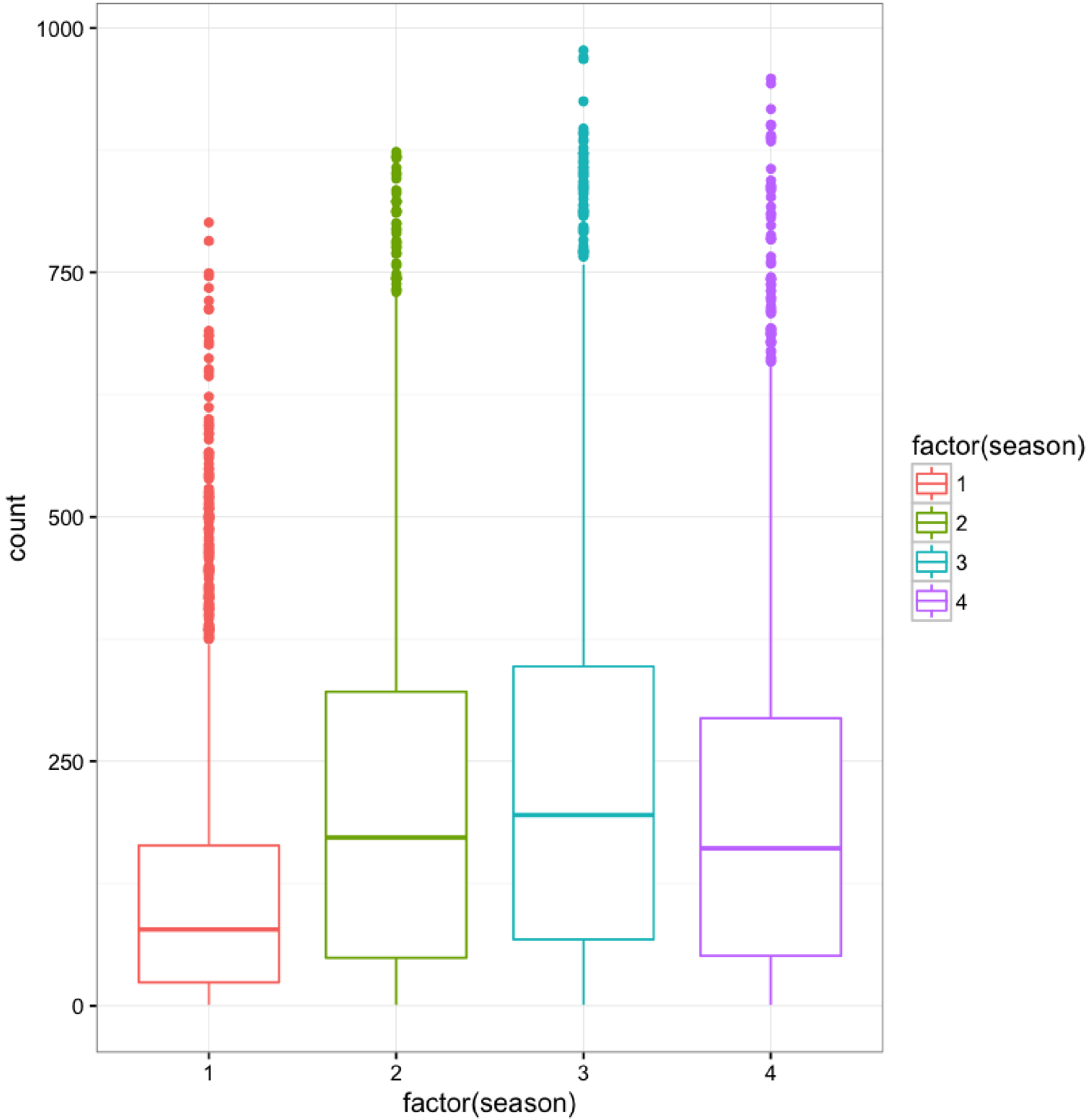
In [32]:
```
cor(bike[,c('temp','count')])
```

Out[32]:

	temp	count
temp	1.0000000	0.3944536
count	0.3944536	1.0000000

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

In [115]:
```
ggplot(bike,aes(factor(season),count)) + geom_boxplot(aes(color=factor(
season))) +theme_bw()
```



Notice what this says:

- A line can't capture a non-linear relationship.
- There are more rentals in winter than in spring

We know of these issues because of the growth of rental count, this isn't due to the actual season!

Feature Engineering

A lot of times you'll need to use domain knowledge and experience to engineer and create new features. Let's go ahead and engineer some new features from the datetime column.

Create an "hour" column that takes the hour from the datetime column. You'll probably need to apply some function to the entire datetime column and reassign it. Hint:

```
time.stamp <- bike$datetime[4]
format(time.stamp, "%H")
```

In [60]:
```
bike$hour <- sapply(bike$datetime,function(x){format(x,"%H")})
```

In [61]:
```
head(bike)
```

Out[61]:

	datetime	season	holiday	workingday	weather	temp	atemp	humidity	windspeed	c
1	2011-01-01	1	0	0	1	9.84	14.395	81	0	3
2	2011-01-01 01:00:00	1	0	0	1	9.02	13.635	80	0	8

	datetime	season	holiday	workingday	weather	temp	atemp	humidity	windspeed	ca
3	2011-01-01 02:00:00	1	0	0	1	9.02	13.635	80	0	5
4	2011-01-01 03:00:00	1	0	0	1	9.84	14.395	75	0	3
5	2011-01-01 04:00:00	1	0	0	1	9.84	14.395	75	0	0
6	2011-01-01 05:00:00	1	0	0	2	9.84	12.88	75	6.0032	0

Now create a scatterplot of count versus hour, with color scale based on temp. Only use bike data where workingday==1.

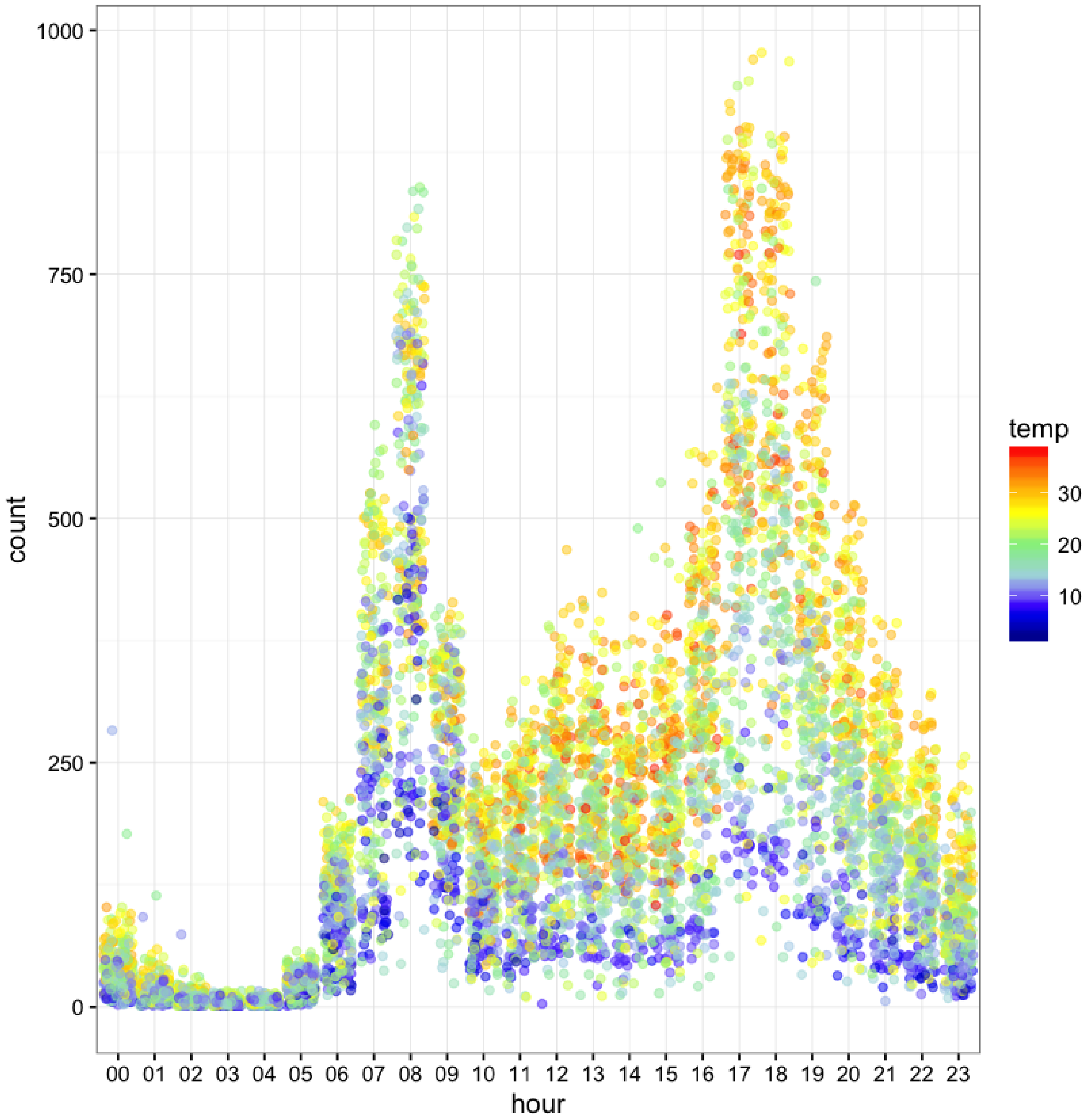
Optional Additions:

- **Use the additional layer: scale_color_gradientn(colors=c('color1',color2,etc..)) where the colors argument is a vector gradient of colors you choose, not just high and low.**
- **Use position=position_jitter(w=1, h=0) inside of geom_point() and check out what it does.**

In [78]:
```
library(dplyr)
```

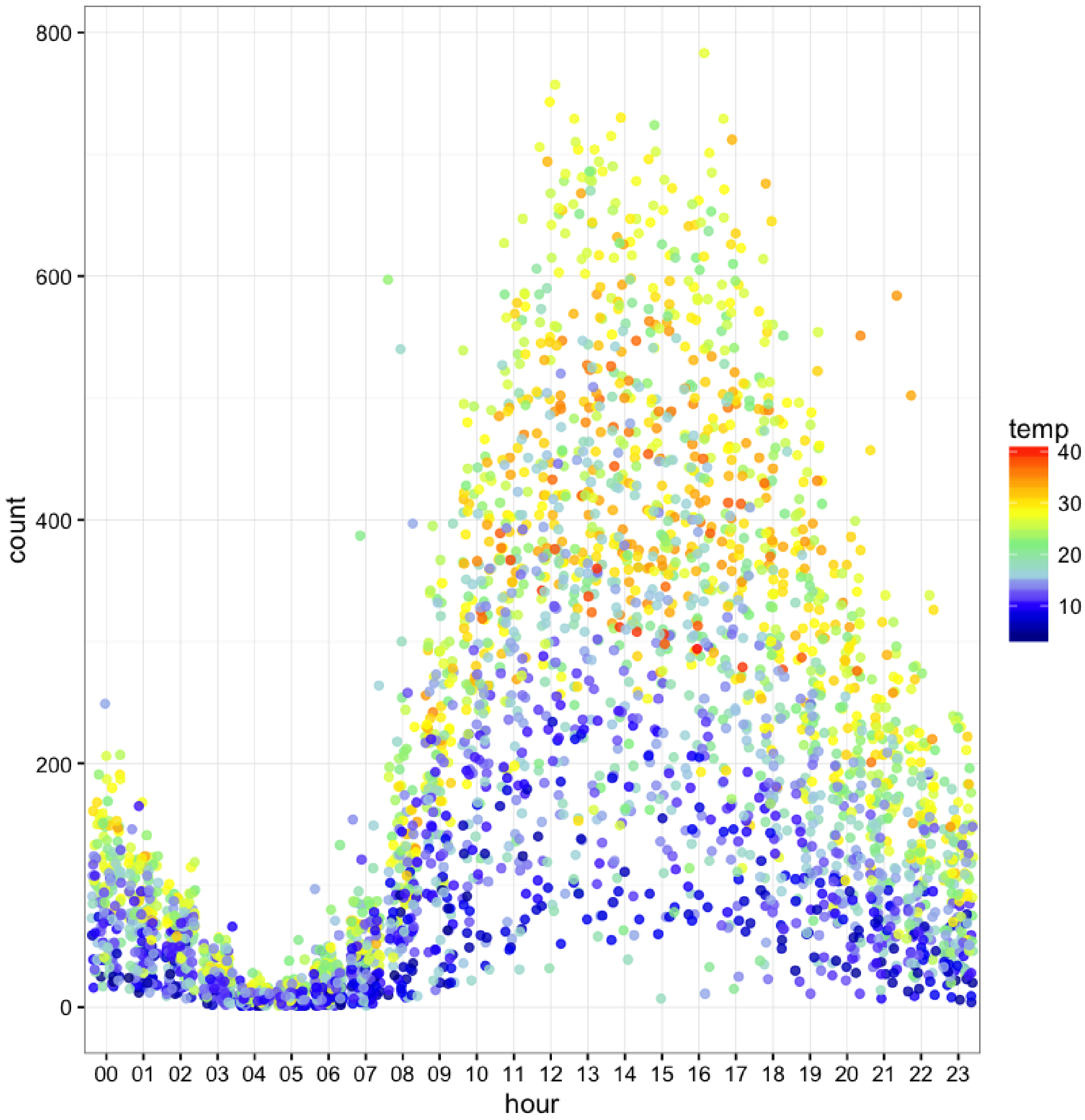
In [103]:
```
pl <- ggplot(filter(bike,workingday==1),aes(hour,count))
pl <- pl + geom_point(position=position_jitter(w=1, h=0),aes(color=temp
),alpha=0.5)
pl <- pl + scale_color_gradientn(colours = c('dark blue','blue','light
```

```
 blue','light green','yellow','orange','red'))
pl + theme_bw()
```



Now create the same plot for non working days:

In [104]:
```
pl <- ggplot(filter(bike,workingday==0),aes(hour,count))
pl <- pl + geom_point(position=position_jitter(w=1, h=0),aes(color=temp
),alpha=0.8)
pl <- pl + scale_color_gradientn(colours = c('dark blue','blue','light
 blue','light green','yellow','orange','red'))
pl + theme_bw()
```



You should have noticed that working days have peak activity during the morning (~8am)

and right after work gets out (~5pm), with some lunchtime activity. While the non-work days have a steady rise and fall for the afternoon

Now let's continue by trying to build a model, we'll begin by just looking at a single feature.

Building the Model

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

In [105]:
```
temp.model <- lm(count~temp,bike)
```

Get the summary of the temp.model

In [107]:
```
summary(temp.model)
```

Out[107]:
```
Call:
lm(formula = count ~ temp, data = bike)

Residuals:
    Min      1Q  Median      3Q     Max 
-293.32 -112.36  -33.36   78.98  741.44 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept)   6.0462     4.4394   1.362    0.173    
temp          9.1705     0.2048  44.783   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 166.5 on 10884 degrees of freedom
Multiple R-squared:  0.1556,	Adjusted R-squared:  0.1555 
F-statistic:  2006 on 1 and 10884 DF,  p-value: < 2.2e-16
```

You should have gotten 6.0462 as the intercept and 9.17 as the temp coeffecient. How can

you interpret these values? Do some wikipedia research, re-read ISLR, or revisit the Linear Regression lecture notebook for more on this.

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.

How many bike rentals would we predict if the temperature was 25 degrees Celsius? Calculate this two ways:

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

You should get around 235.3 bikes.

In [108]:
```
# Method 1
6.0462 + 9.17*25
```

Out[108]: 235.2962

In [121]:
```
# Method 2
temp.test <- data.frame(temp=c(25))
```

```
predict(temp.model,temp.test)
```

Out[121]: **1:** 235.309724995272

Use sapply() and as.numeric to change the hour column to a column of numeric values.

In [122]:
```
bike$hour <- sapply(bike$hour,as.numeric)
```

Finally build a model that attempts to predict count based off of the following features. Figure out if theres a way to not have to pass/write all these variables into the lm() function. Hint: StackOverflow or Google may be quicker than the documentation.

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

In [127]:
```
model <- lm(count ~ . -casual - registered -datetime -atemp,bike )
```

Get the summary of the model

In [128]:
```
summary(model)
```

Out[128]:
```
Call:
lm(formula = count ~ . - casual - registered - datetime - atemp,
    data = bike)

Residuals:
    Min      1Q  Median      3Q     Max
-324.61  -96.88  -31.01   55.27  688.83
```

```
Coefficients:
             Estimate Std. Error t value Pr(>|t|)    
(Intercept)  46.91369    8.45147   5.551 2.91e-08 ***
season       21.70333    1.35409  16.028  < 2e-16 ***
holiday     -10.29914    8.79069  -1.172    0.241    
workingday   -0.71781    3.14463  -0.228    0.819    
weather      -3.20909    2.49731  -1.285    0.199    
temp          7.01953    0.19135  36.684  < 2e-16 ***
humidity     -2.21174    0.09083 -24.349  < 2e-16 ***
windspeed     0.20271    0.18639   1.088    0.277    
hour          7.61283    0.21688  35.102  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 147.8 on 10877 degrees of freedom
Multiple R-squared:  0.3344,	Adjusted R-squared:  0.3339 
F-statistic:   683 on 8 and 10877 DF,  p-value: < 2.2e-16
```

Did the model perform well on the training data? What do you think about using a Linear Model on this data?

A linear model like the one we chose which uses OLS won't be able to take into account seasonality of our data, and will get thrown off by the growth in our dataset, accidentally attributing it towards the winter season, instead of realizing its just overall demand growing! Later on, we'll see if other models may be a better fit for this sort of data.

You should have noticed that this sort of model doesn't work well given our seasonal and time series data. We need a model that can account for this type of trend, read about Regression Forests for more info if you're interested! For now, let's keep this in mind as a learning experience and move on towards classification with Logistic Regression!

Optional: See how well you can predict for future data points by creating a train/test split. But instead of a random split, your split should be "future" data for test, "previous" data for train.