New Dehli Weather Portfolio Project

Olivia Lund

This project is an analysis of a dataset of detailed weather records from New Dehli, India, starting in November 1996 and ending in April 2014. This dataset was chosen because of its size and detail, featuring over one hundred thousand entries and over twenty different fields. These data can be used to perform a variety of analyses linking different attributes, for example finding what times of the year are more prone to smog, or if wind direction can be a good indicator for presence of precipitation.

Before being able to conduct any effective data analysis, as with any other skilled project, the right tools have to be selected. This project will be using the library TidyVerse because of its extensive data organization and the library GGPlot because of its consice and dynamic visualization options.

```
library("tidyverse")
```

```
## ── Attaching packages ──────────────────────────────────────────────────────
── tidyverse 1.3.0 ──
```

```
## ✔ ggplot2 3.2.1     ✔ purrr   0.3.3
## ✔ tibble  2.1.3     ✔ dplyr   0.8.3
## ✔ tidyr   1.0.0     ✔ stringr 1.4.0
## ✔ readr   1.3.1     ✔ forcats 0.4.0
```

```
## ── Conflicts ─────────────────────────────────────────────────────────── tid
yverse_conflicts() ──
## ✖ dplyr::filter() masks stats::filter()
## ✖ dplyr::lag()    masks stats::lag()
```

```
library("ggplot2")
```

In order to read and modify the dataset, it first has to be read into this workspace, done here via local filepath. This data is sourced from user mahirkukreja who posted it to kaggle.com on the page https://www.kaggle.com/mahirkukreja/delhi-weather-data/ (https://www.kaggle.com/mahirkukreja/delhi-weather-data/).

```
weather <- read.csv(file="https://raw.githubusercontent.com/kippyan/DataScience/master/d
ehliweather.csv",)
colnames(weather)
```

```
##  [1] "datetime_utc" "X_conds"      "X_dewptm"     "X_fog"        "X_hail"
##  [6] "X_heatindexm" "X_hum"        "X_precipm"    "X_pressurem"  "X_rain"
## [11] "X_snow"       "X_tempm"      "X_thunder"    "X_tornado"    "X_vism"
## [16] "X_wdird"      "X_wdire"      "X_wgustm"     "X_windchillm" "X_wspdm"
```

This readout of the column labels shows that the original label format is strange and confusing at times, and uses strange or arbitrary notations and abbreviations. In order to help make the dataset tidy and readable, some of these names will be changed to be more clear:

```
colnames(weather)[colnames(weather)=="datetime_utc"]<-"datetime_utc"
colnames(weather)[colnames(weather)=="X_conds"]<-"air_condition"
colnames(weather)[colnames(weather)=="X_dewptm"]<-"dew_point"
colnames(weather)[colnames(weather)=="X_fog"]<-"any_fog"
colnames(weather)[colnames(weather)=="X_hail"]<-"any_hail"
colnames(weather)[colnames(weather)=="X_heatindexm"]<-"heat_index"
colnames(weather)[colnames(weather)=="X_hum"]<-"humidity"
colnames(weather)[colnames(weather)=="X_precipm"]<-"precipitation"
colnames(weather)[colnames(weather)=="X_pressurem"]<-"air_pressure"
colnames(weather)[colnames(weather)=="X_rain"]<-"any_rain"
colnames(weather)[colnames(weather)=="X_snow"]<-"any_snow"
colnames(weather)[colnames(weather)=="X_tempm"]<-"temperature"
colnames(weather)[colnames(weather)=="X_thunder"]<-"any_thunder"
colnames(weather)[colnames(weather)=="X_tornado"]<-"any_tornadoes"
colnames(weather)[colnames(weather)=="X_vism"]<-"visibility"
colnames(weather)[colnames(weather)=="X_wdird"]<-"wind_angle"
colnames(weather)[colnames(weather)=="X_wdire"]<-"wind_direction"
colnames(weather)[colnames(weather)=="X_wgustm"]<-"wind_gust_speed"
colnames(weather)[colnames(weather)=="X_windchillm"]<-"wind_chill"
colnames(weather)[colnames(weather)=="X_wspdm"]<-"wind_average_speed"
head(weather, n =5)
```

```
##     datetime_utc air_condition dew_point any_fog any_hail heat_index humidity
## 1 19961101-11:00         Smoke         9       0        0         NA       27
## 2 19961101-12:00         Smoke        10       0        0         NA       32
## 3 19961101-13:00         Smoke        11       0        0         NA       44
## 4 19961101-14:00         Smoke        10       0        0         NA       41
## 5 19961101-16:00         Smoke        11       0        0         NA       47
##   precipitation air_pressure any_rain any_snow temperature any_thunder
## 1            NA         1010        0        0          30           0
## 2            NA        -9999        0        0          28           0
## 3            NA        -9999        0        0          24           0
## 4            NA         1010        0        0          24           0
## 5            NA         1011        0        0          23           0
##   any_tornadoes visibility wind_angle wind_direction wind_gust_speed wind_chill
## 1             0        5.0        280           West              NA         NA
## 2             0         NA          0          North              NA         NA
## 3             0         NA          0          North              NA         NA
## 4             0        2.0          0          North              NA         NA
## 5             0        1.2          0          North              NA         NA
##   wind_average_speed
## 1                7.4
## 2                 NA
## 3                 NA
## 4                 NA
## 5                0.0
```

These data labels make the data easier to analyze by clearly describing what they represent. It requires no experience with data sience to understand what each of these values is generally trying to say.

Specifics on each field will be explained below, which will give us an idea what fields warrant further changes:

- datetime_utc: This is simply the date and time of day using the Coordinated Universal Time system. It is continuous and should be changed to the date format built into R.
- air_condition: This field contains the primary air condition at the time of the reading. It is a categorical form variable that can take one of many different types of values. Here are the first ten encountered in the list:
  - [1] Blowing Sand
  - [2] Clear
  - [3] Haze
  - [4] Unknown
  - [5] Scattered Clouds
  - [6] Shallow Fog
  - [7] Mostly Cloudy
  - [8] Fog
  - [9] Partly Cloudy
  - [10] Patches of Fog
- dew_point: This is the dew point at the current time, which is the temperature at which dew will form, given the current temperature and pressure conditions. It is a decimal with one point of precision.
- any_fog: This attribute represents whether or not there currently is fog, represented as a boolean value.
- any_hail: This is a boolean attribute representing if there currently is hail. There are no data points during which it was hailing, making this a prime candidate for removal.
- heat_index: This is a continuous number variable representing the current heat index, which takes into account humidity to more accurately represent percieved temperature.
- humidity: This is a continuous decimal number variable that represents the current humidity.
- precipitiaton: All entries for this value contain missing entries, making it a prime candidate for removal.
- air_pressure: This value represents air pressure. The values are all around 1000, which isn’t in a reasonable range for any of the most common units, but this variable could still be kept around to compare relative air pressure within the dataset.
- any_rain: A boolean value expressing the presence of rain.
- any_snow: A boolean value expressing the presence of snow. Only true for one single reading in May of 2014. Being virtually empty makes it a candidate for removal.
- temperature: A decimal number representing the current temperature in centigrade.
- any_thunder: A boolean value expressing the presence of thunder.
- any_tornadoes: A boolean value expressing the presence of tornadoes. Only true for two entries, one during August of 1999, and one during February of 2007. Curiously, snow is less common in New Delhi than tornadoes. Because this is mostly empty, it is a candidate for deletion.
- visibility: A decimal number representing the current visibility, in km.
- wind_angle: An integer representing the degrees to the right of north that the wind is blowing.
- wind_direction: A string representing the direction of the wind in 3-letter cardinal direction combinations.
- wind_gust_speed: A decimal representing the max speed in km/hr that the wind is blowing in periods of fluctuation.
- wind_chill: An integer representing the apparent temperature due to the speed of the wind in centigrade.
- wind_average_speed: A decimal representing the current average speed that the wind is going.

Now that we have looked at our data, we can modify it and clean it up.

```
weather <- subset(weather, select = -c(any_snow, any_hail, any_tornadoes, precipitatio
n))
weather <- subset(weather, !is.na(weather$heat_index))
#weather <- subset(weather, !is.na(weather$wind_chill))
#weather <- subset(weather, !is.na(weather$wind_gust_speed))

weather <- subset(weather, !(weather$air_pressure == -9999))
weather$datetime_utc[1]
```

```
## [1] 19961103-11:00
## 100990 Levels: 19961101-11:00 19961101-12:00 19961101-13:00 ... 20170424-18:00
```

To make the data easier to work with and combine with other datasets, we can first parse the date and time into R's datetime format, and then break it up into a column for date and a column for time.

```
weather$time = weather$datetime_utc
weather <- weather[,c(1,17,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16)]
weather$time <- strsplit(as.character(weather$time),"-")
weather$time <- sapply(weather$time, function(time){ return(time[2]) })
weather$time <- parse_time(as.character(weather$time), format = "%H:%M")
head(weather$time)
```

```
## 11:00:00
## 06:00:00
## 09:00:00
## 11:00:00
## 12:00:00
## 11:00:00
```

Unfortunately, this appears to only be capable of producing datetime objects with the default date, so we might as well just keep a datetime object instead, and use this new column as a date column, since using date will be useful in the future.

```
weather$date = weather$datetime_utc
weather <- weather[,c(1,2,18,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17)]
weather$date <- strsplit(as.character(weather$datetime_utc),"-")
weather$date <- sapply(weather$date, function(date){ return(date[1]) })
weather$date <- parse_date(as.character(weather$date), format = "%Y%m%d")
head(weather$date)
```

```
## [1] "1996-11-03" "1996-11-04" "1996-11-13" "1996-11-13" "1997-03-01"
## [6] "1997-03-06"
```

And as we can see, the date column has been created correctly. Now, we can go ahead and parse the datetime_utc column into an R datetime object

```
weather$datetime_utc <- as.POSIXct(parse_datetime(as.character(weather$datetime_utc), fo
rmat = "%Y%m%d-%H:%M"))
head(weather$datetime_utc)
```

```
## [1] "1996-11-03 11:00:00 UTC" "1996-11-04 06:00:00 UTC"
## [3] "1996-11-13 09:00:00 UTC" "1996-11-13 11:00:00 UTC"
## [5] "1997-03-01 12:00:00 UTC" "1997-03-06 11:00:00 UTC"
```

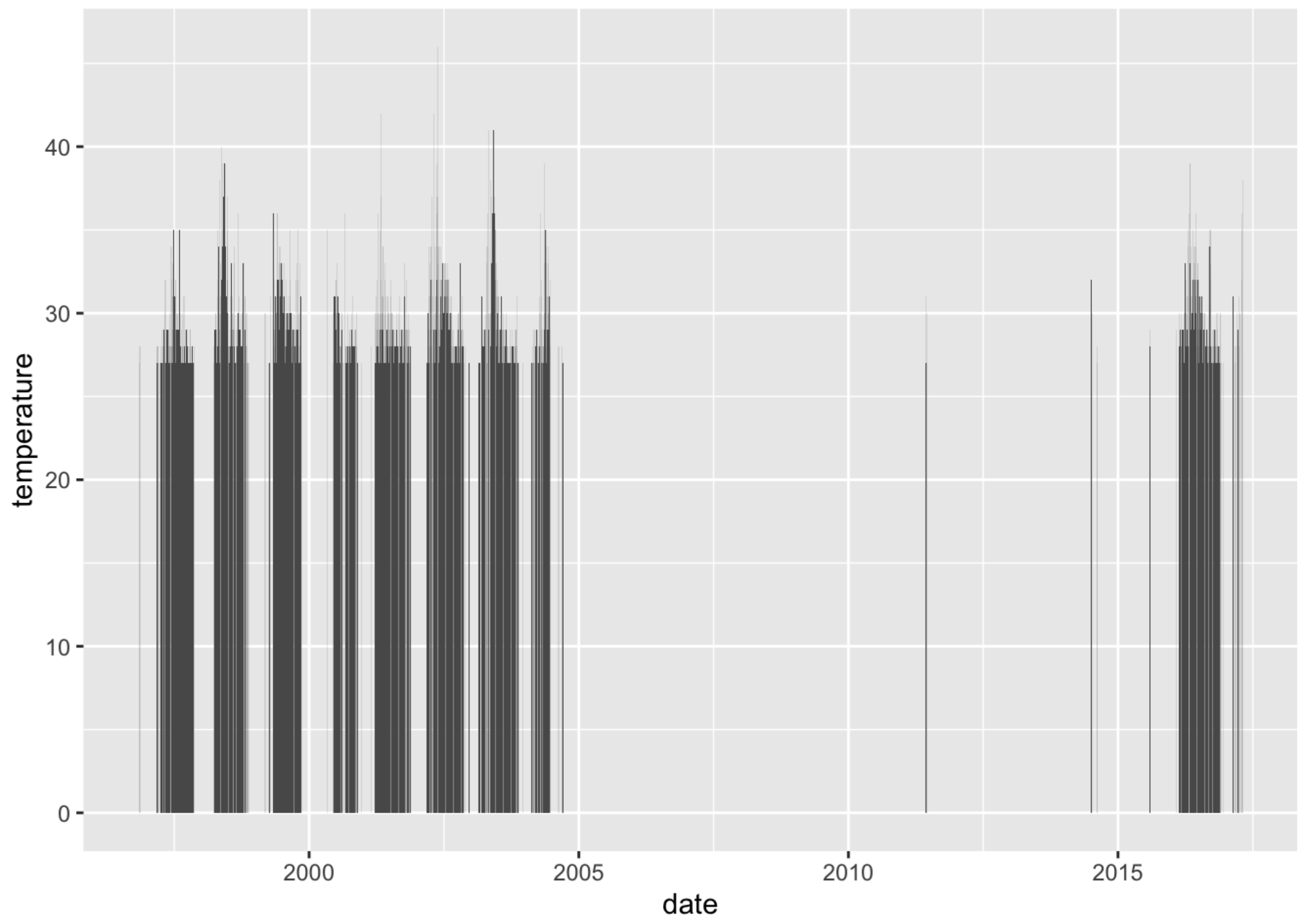
```
dailyweather <- weather %>% arrange(date, time) %>% group_by(date) %>% slice(1)
head(dailyweather$datetime_utc)
```

```
## [1] "1996-11-03 11:00:00 UTC" "1996-11-04 06:00:00 UTC"
## [3] "1996-11-13 09:00:00 UTC" "1997-03-01 12:00:00 UTC"
## [5] "1997-03-06 11:00:00 UTC" "1997-03-07 08:00:00 UTC"
```

Most of the major changes are now out of the way for exploring the dataset, and we can now begin with visualization.

To begin, a simple graph of temperature against date, to see if there are any appreciable effects over time.

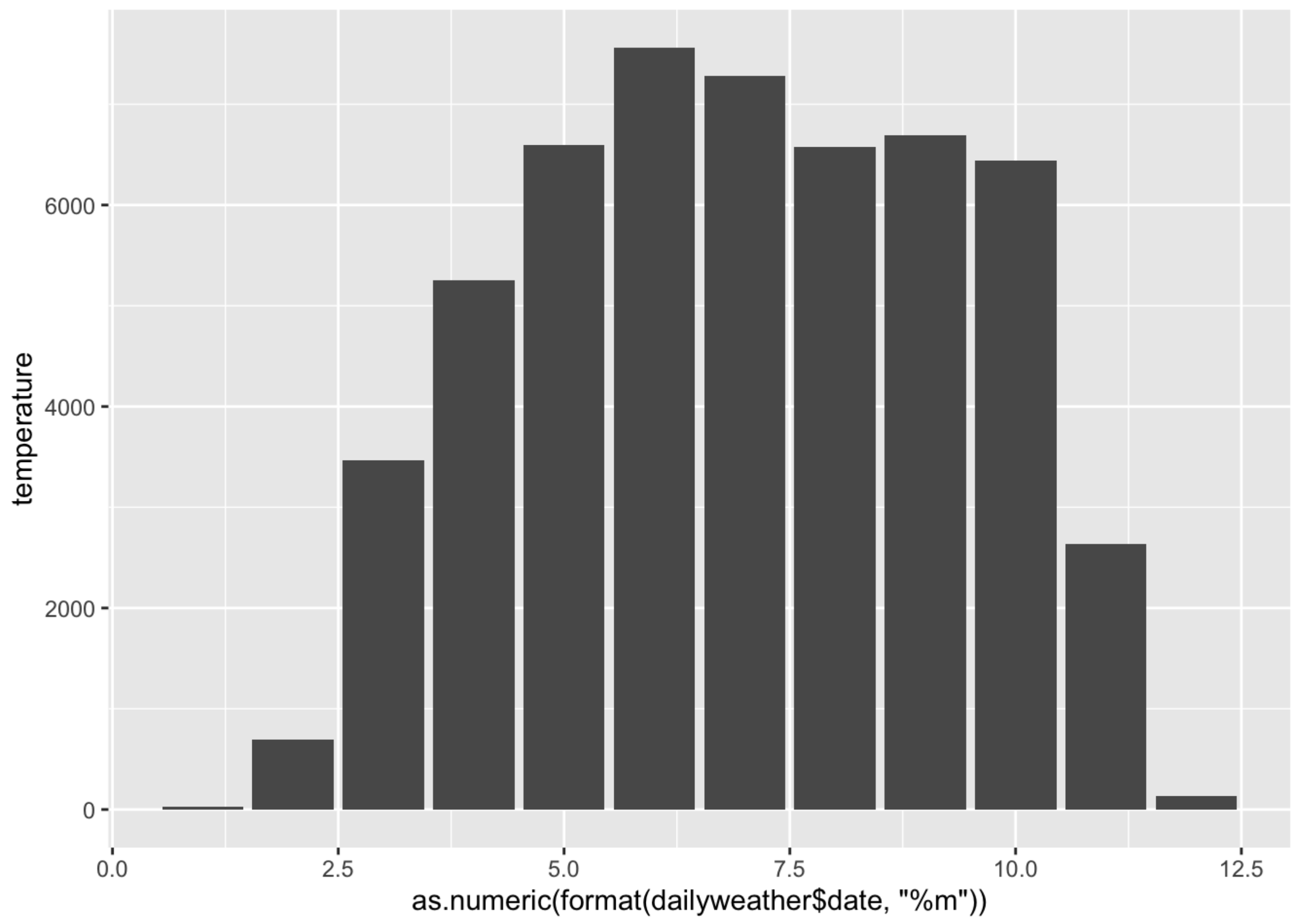
```
ggplot(dailyweather, aes(date, temperature)) + geom_col()
```



If there are any appreciable changes in overall temperature over the years, it's very difficult to see. This graph best illustrates the large gap in the data.

It would be very interesting to see a visualization of various statistics by month over the course of a year, however it appears quite difficult in R, which vectorizes attributes of the data frame in ways that can be hard to deal with In this case, it's summing all of the temperatures in that month over the course of an entire year.

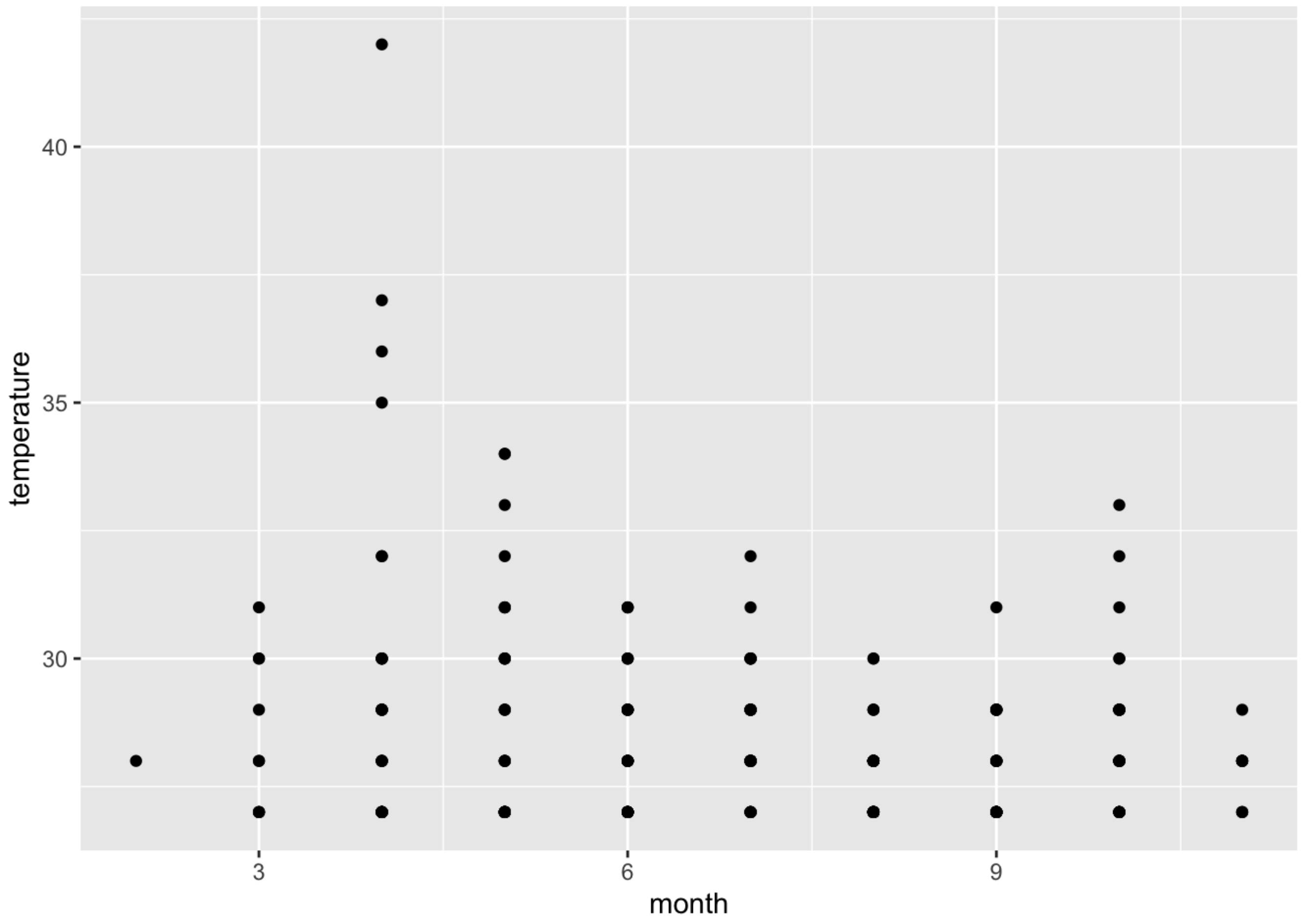
```
ggplot(dailyweather, aes(as.numeric(format(dailyweather$date, "%m")), temperature)) + ge
om_col()
```



It’s not very helpful to have a graph that prints the sum of all daily temperatures in each month.

To look at a specific year, we can do the following:

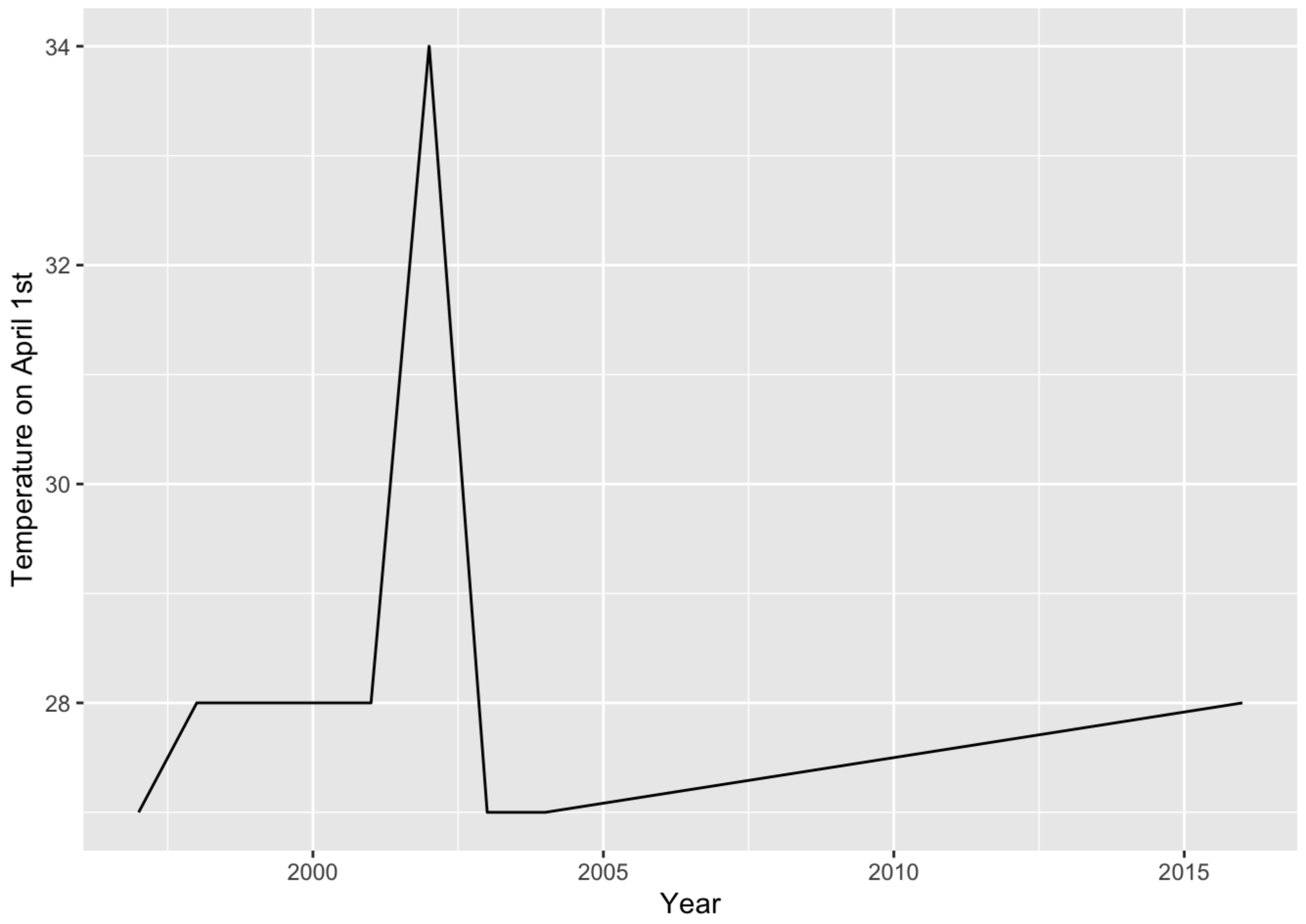
```
tempdata = subset(dailyweather, format.Date(date, "%Y") =="2001" )
ggplot(tempdata, aes(as.numeric(format(tempdata$date, "%m")), temperature)) + geom_point
() + labs(x = "month")
```



This graphic is much less impressive but till communicates the temperature correctly. The temperature reaches higher levels during a portion of the year, as one would expect.

Now, to do the same thing with a specific day across all the years for which data is available:

```
tempdata = subset(dailyweather, format.Date(date, "%m") =="04" & format.Date(date, "%d")
=="01")
ggplot(tempdata, aes(as.numeric(format(tempdata$date, "%Y")), temperature)) + geom_path
() + labs(x = "Year", y = "Temperature on April 1st")
```



This would be very interesting if the data were more dense, but unfortunately it's hard to make any interesting trends from this.

This project holds a lot of potential for identifying relationships between many different aspects of the climate of New Delhi at various points throughout the past twenty years, and displaying the power of Data Science in the process.