Google Data Analytics Case Study

Project Overview

In this case study, I play the position of a junior data analyst for Bellabeat, a high-tech manufacturer of women's health products. The stakeholders of the company believes that analyzing non-Bellabeat smart device fitness data could help unlock new growth opportunities for the company. I've been entrusted with focusing on one of Bellabeat's products and analyzing non-Bellabeat smart device data to gain insight into how consumers use their non-Bellabeat smart devices and how these trends can be applied to Bellabeat customers. I'll be using data from FitBit Fitness Tracker, a public dataset made available by Mobius in Kaggle. I am going to provide the findings of my research to the Bellabeat executive team, along with my high-level recommendations that will them with their marketing efforts.

Business Task

Analyzing non-Bellabeat smart device data to gain insight into how consumers are using their non-Bellabeat smart devices.

Goals

- Find trends among the people using their non-Bellabeat smart devices
- See if these trends align with Bellabeat users
- Use that knowledge to make a marketing strategy

Data Sources

For our task we are using FitBit Fitness Tracker Data (https://www.kaggle.com/datasets/arashnic/fitbit). This Kaggle data set contains personal fitness tracker from thirty fitbit users over the course of a month. Thirty eligible Fitbit users consented to the submission of personal tracker data, including minute-level output for physical activity, calories burned, weight and sleep monitoring. It includes information can be used to explore users' habits.

Cleaning And Manipulation Of Data

Using R as the data analysis tool for the project

Loading packages

```
library(tidyverse)
```

```
## ── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──
## ✔ dplyr     1.1.4     ✔ readr     2.1.5
## ✔ forcats   1.0.0     ✔ stringr   1.5.1
## ✔ ggplot2   3.5.0     ✔ tibble    3.2.1
## ✔ lubridate 1.9.3     ✔ tidyr     1.3.1
## ✔ purrr     1.0.2
## ── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ──
## ✖ dplyr::filter() masks stats::filter()
## ✖ dplyr::lag()    masks stats::lag()
## ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become
errors
```

Importing csv files and creating data sets

```
daily_activity <- read.csv("C:\\Users\\1iCE\\Desktop\\New folder (2)\\daily_data\\dailyActivity_merged.csv")
daily_weight <- read.csv("C:\\Users\\1iCE\\Desktop\\New folder (2)\\daily_data\\weightLogInfo_merged.csv")
daily_sleep <- read.csv("C:\\Users\\1iCE\\Desktop\\New folder (2)\\daily_data\\sleepDay_merged.csv")
```

Showing first few data on the data sets

```
head(daily_activity)
```

	Id	ActivityDate	TotalSteps	TotalDistance	TrackerDistance	LoggedActivitiesDistance
	<dbl>	<chr>	<int>	<dbl>	<dbl>	<dbl>
1	1503960366	4/12/2016	13162	8.50	8.50	0
2	1503960366	4/13/2016	10735	6.97	6.97	0
3	1503960366	4/14/2016	10460	6.74	6.74	0
4	1503960366	4/15/2016	9762	6.28	6.28	0
5	1503960366	4/16/2016	12669	8.16	8.16	0
6	1503960366	4/17/2016	9705	6.48	6.48	0

6 rows | 1-7 of 16 columns

```
head(daily_weight)
```

	Id	Date	Weight…	WeightPounds	F..	BMI	IsManualReport	LogId
	<dbl>	<chr>	<dbl>	<dbl>	<int>	<dbl>	<chr>	<dbl>
1	1503960366	5/2/2016 11:59:59 PM	52.6	115.9631	22	22.65	True	1.462234e+12
2	1503960366	5/3/2016 11:59:59 PM	52.6	115.9631	*NA*	22.65	True	1.462320e+12
3	1927972279	4/13/2016 1:08:52 AM	133.5	294.3171	*NA*	47.54	False	1.460510e+12
4	2873212765	4/21/2016 11:59:59 PM	56.7	125.0021	*NA*	21.45	True	1.461283e+12
5	2873212765	5/12/2016 11:59:59 PM	57.3	126.3249	*NA*	21.69	True	1.463098e+12
6	4319703577	4/17/2016 11:59:59 PM	72.4	159.6147	25	27.45	True	1.460938e+12

6 rows

```
head(daily_sleep)
```

	Id	SleepDay	TotalSleepRecords	TotalMinutesAsleep	TotalTimeInBed
	<dbl>	<chr>	<int>	<int>	<int>
1	1503960366	4/12/2016 12:00:00 AM	1	327	346
2	1503960366	4/13/2016 12:00:00 AM	2	384	407
3	1503960366	4/15/2016 12:00:00 AM	1	412	442

	Id	SleepDay	TotalSleepRecords	TotalMinutesAsleep	TotalTimeInBed
	<dbl>	<chr>	<int>	<int>	<int>
4	1503960366	4/16/2016 12:00:00 AM	2	340	367
5	1503960366	4/17/2016 12:00:00 AM	1	700	712
6	1503960366	4/19/2016 12:00:00 AM	1	304	320

6 rows

Formatting the data sets

```
#Formatting rows of the date column to the standard date format
daily_activity$ActivityDate <- as.Date(daily_activity$ActivityDate, "%m/%d/%Y")
daily_weight$Date <- as.Date(daily_weight$Date, "%m/%d/%Y")
daily_sleep$SleepDay <- as.Date(daily_sleep$SleepDay, "%m/%d/%Y")

#Changing column name to use "Date" as a foreign key
colnames(daily_activity)[2] <- "Date"
colnames(daily_weight)[2] <- "Date"
colnames(daily_sleep)[2] <- "Date"
```

Making copy of the data sets to avoid unintentional changes on original data sets

```
#coping data sets
activity <- daily_activity
weight <- daily_weight
sleep <- daily_sleep
```

Merging data sets

```
#Merging
activity_weight <- merge(x=activity, y=weight)
activity_sleep <- merge(x=activity, y=sleep)
```

Analysis

Organizing and categorizing users based on the duration of their sleep

Types of sleep:

- Normal Sleepers: Sleep time is between 360 minutes (6 hours) to 480 minutes (8 hours).
- Bad Sleepers: Sleep times is less then 360 minutes (6 hours).
- Over Sleepers: Sleep time is over 480 minutes (8 hours).

```
#Labeling users according to their sleeping time
sleep_type <- sleep %>%
  mutate(OverSlept = case_when(TotalMinutesAsleep > 480 ~ 1, TRUE ~ 0),
         UnderSlept = case_when(TotalMinutesAsleep < 360 ~ 1, TRUE ~ 0),
         Normal = case_when(TotalMinutesAsleep > 360 & TotalMinutesAsleep < 480 ~ 1, TRUE ~ 0)) %
>%
  group_by(Id) %>%
  summarise(OverSlept = sum(OverSlept),
            UnderSlept = sum(UnderSlept),
            Normal = sum(Normal)) %>%
  mutate(SleepType = case_when(OverSlept > UnderSlept & OverSlept > Normal ~ "Over Sleeper",
                               UnderSlept > OverSlept & UnderSlept > Normal ~ "Bad Sleeper",
                               Normal > OverSlept & Normal > UnderSlept ~ "Normal Sleeper")) %>% d
rop_na()
```

Organizing and categorizing users based on their calories burned and distance covered

```
activity_type_length <- activity %>%
  group_by(Id) %>%
  summarise(Calories = sum(Calories))%>%
  mutate(ActiveType = case_when(Calories > quantile(Calories, probs = 0.75) ~ ">75%",
                                Calories > quantile(Calories, probs = 0.50) ~ "50%<",
                                Calories > quantile(Calories, probs = 0.25) ~ ">25%",
                                Calories < quantile(Calories, probs = 0.25) ~ "25%<",)) %>% drop_n
a()

activity_type_distance <- activity %>%
  group_by(Id) %>%
  summarise(TotalDistance = sum(TotalDistance))%>%
  mutate(ActiveType = case_when(TotalDistance > quantile(TotalDistance, probs = 0.75) ~ ">75%",
                                TotalDistance > quantile(TotalDistance, probs = 0.50) ~ "50%<",
                                TotalDistance > quantile(TotalDistance, probs = 0.25) ~ ">25%",
                                TotalDistance < quantile(TotalDistance, probs = 0.25) ~ "25%<",))
%>% drop_na()
```

Formatting and merging data for visualization

```
#Removing irrelevant rows
activity_type_length <- activity_type_length[,-2]
activity_type_distance <- activity_type_distance[,-2]
sleep_type <- sleep_type[,-2:-4]

#Merging data sets
user_calories_burned_sleep_type <- merge(x=activity_type_length, y=sleep_type, by="Id")
user_distance_sleep_type <- merge(x=activity_type_distance, y=sleep_type, by="Id")
```

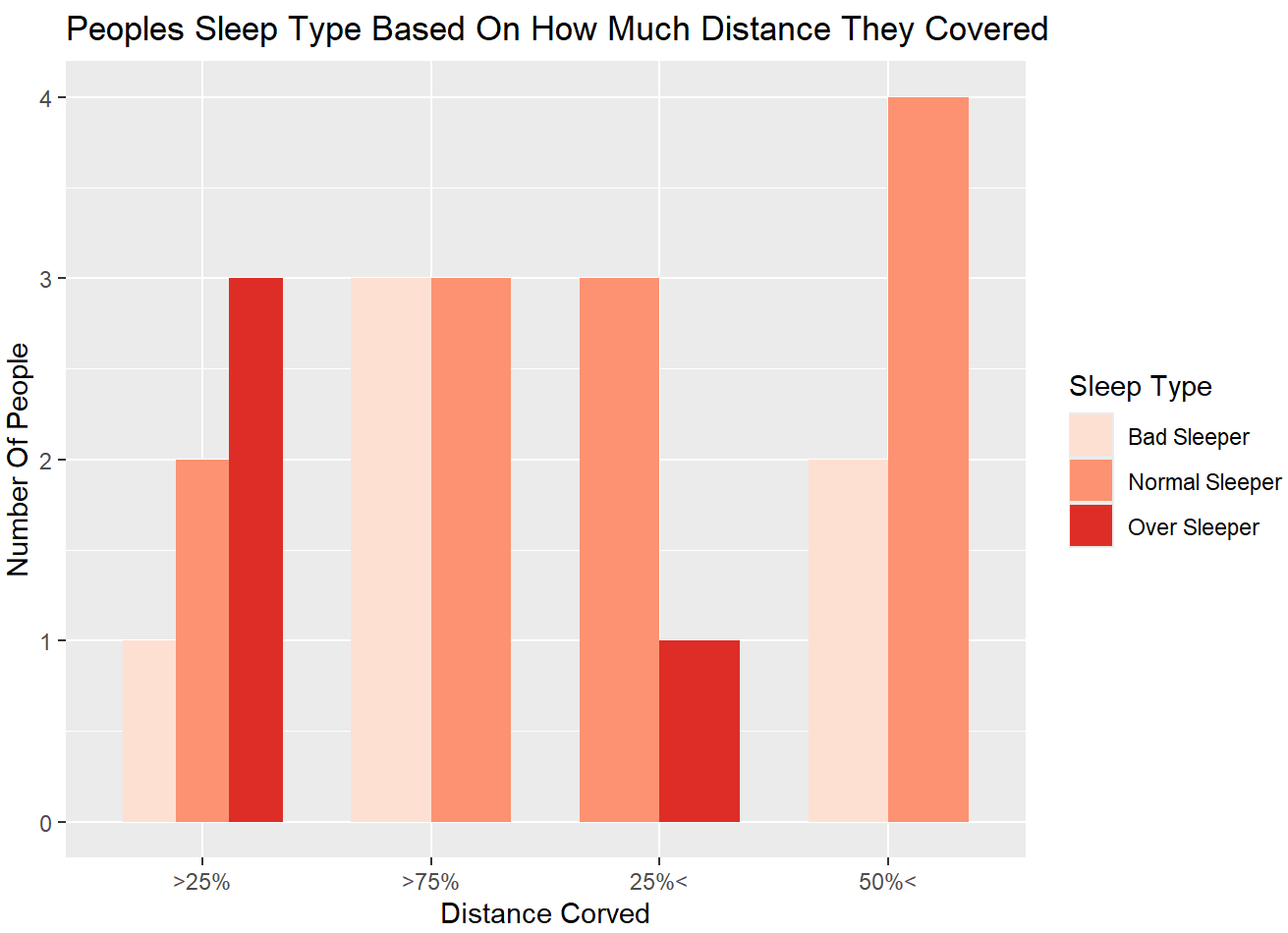
Graphical Represntaion Of The Key Findings

Finding 1 Our first bar graph shows users sleep type based on how much calories they burned where we can see users who are burning more calories are sleeping better. But the number of bad sleepers and over sleepers are still there below the 75th percentile.

```
#Effects of calories burned on sleep type
ggplot(data=user_calories_burned_sleep_type) +
  geom_bar(mapping = aes(x=ActiveType, fill=SleepType), position = position_dodge(), width = 0.7)+
  labs(title = "Peoples Sleep Type Based On How Much Calories They Burned", x="Calories Burned", y
="Number Of People", fill="Sleep Type")+
  scale_fill_brewer(palette="Reds")
```

Finding 2 Our next bar graph shows users sleep type based on how much distance the covered where we can again see users who are covering more distance while walking or jogging are sleeping better. But the discrepancy of bad sleepers and over sleepers are still there after the 75th percentile.

```
#Effects of distance covered on sleep type
ggplot(data=user_distance_sleep_type) +
  geom_bar(mapping = aes(x=ActiveType, fill=SleepType), position = position_dodge(), width = 0.7)+
  labs(title = "Peoples Sleep Type Based On How Much Distance They Covered", x="Distance Corved",
y="Number Of People", fill="Sleep Type")+
  scale_fill_brewer(palette="Reds")
```



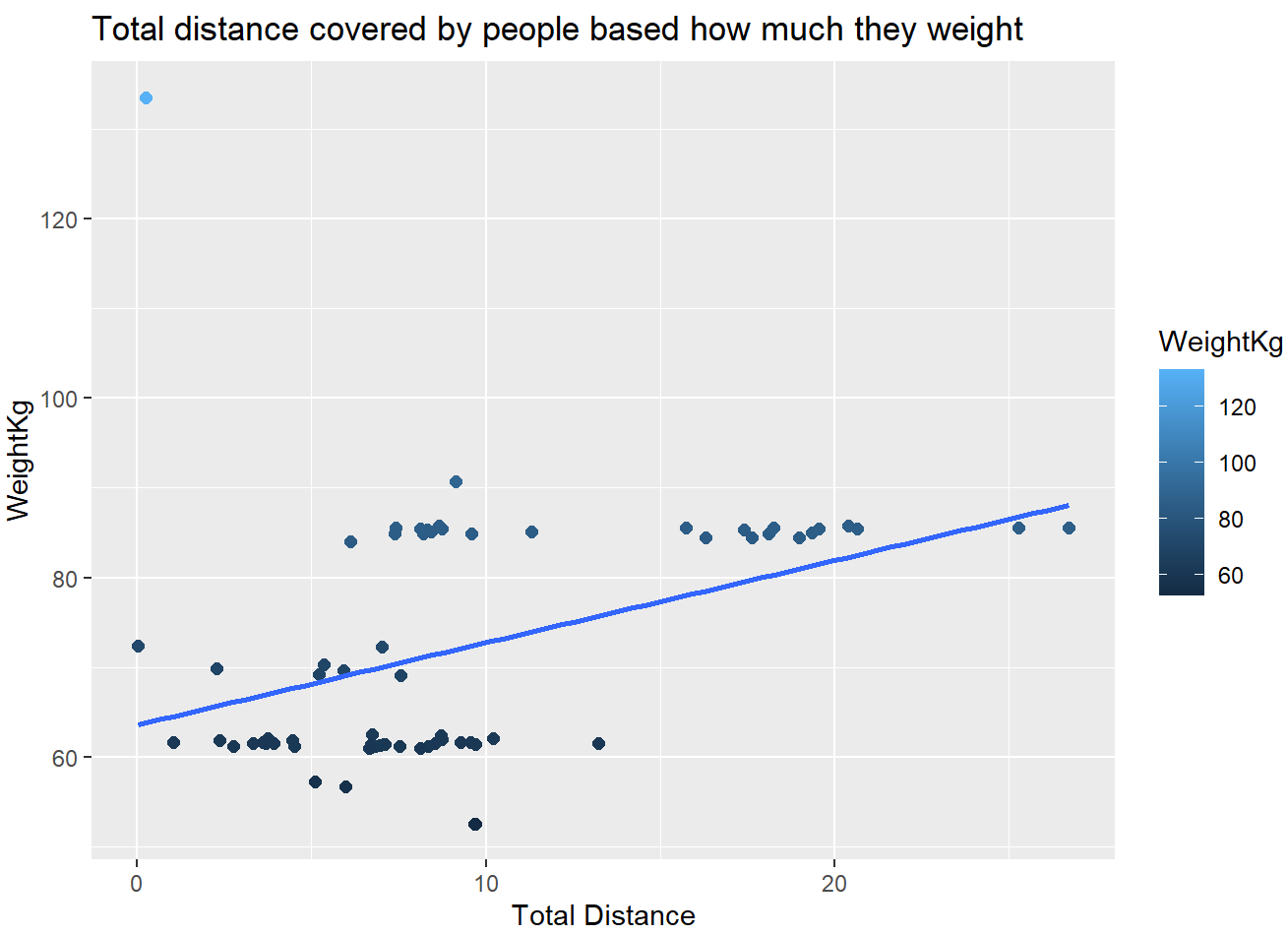
Finding 3 We shed some light on the earlier discrepancy of our first two findings in our next two scatter plots. Here, we can see that users who are over the mean weight are the ones who are burning more calories and covering more distance.

```
#Showing relation between calories burned and weight of the individuals
ggplot(data=activity_weight) +
  geom_point(mapping = aes(x=Calories, y=WeightKg, color=WeightKg), size=2)+
  geom_smooth(aes(x=Calories, y=WeightKg), se = FALSE)+
  labs(title = "Calories burned by people based how much they weight")
```

```
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'
```


```
#Showing relation between distance covered and weight of the individuals
ggplot(data=activity_weight) +
  geom_point(mapping = aes(x=TotalDistance, y=WeightKg, color=WeightKg), size=2)+
  geom_smooth(aes(x=TotalDistance, y=WeightKg), method = lm, se = FALSE)+
  labs(title = "Total distance covered by people based how much they weight", x="Total Distance")
```

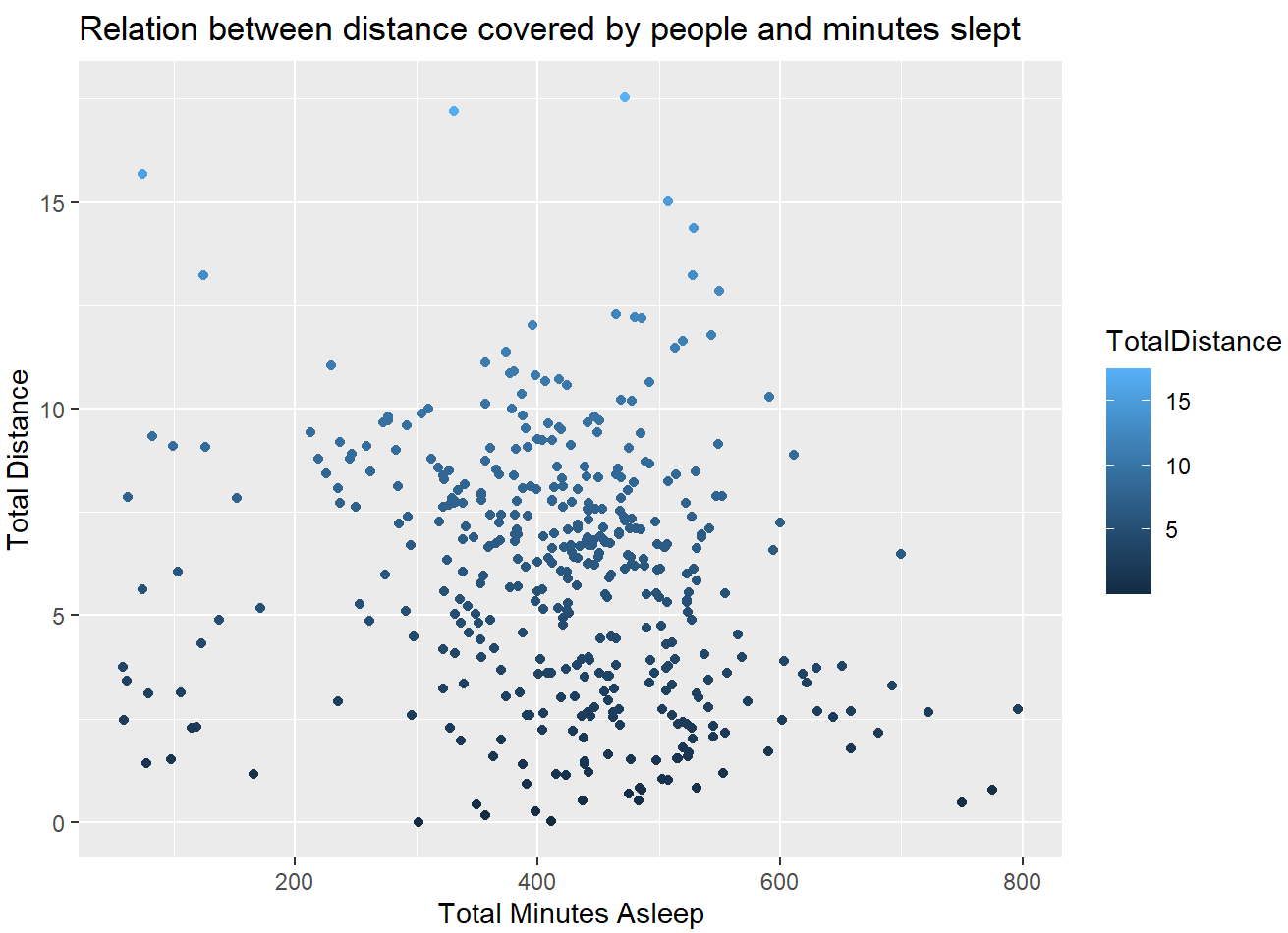
```
## `geom_smooth()` using formula = 'y ~ x'
```



Finding 4 Our last two scatter plot shows users sleep time based on how much calories they burned and distance they covered. Here we can see a clearly that users who are working out more (by burning calories and covering distance) are the ones who are with normal sleep time.

```
#Showing relation between calories burned and minutes slept
ggplot(data=activity_sleep) +
  geom_point(mapping = aes(x=TotalMinutesAsleep, y=Calories, color=Calories))+
  labs(title = "Relation between calories burned by people and minutes slept", x='Total Minutes As
leep')
```


```
#Showing relation between distance covered and minutes slept
ggplot(data=activity_sleep) +
  geom_point(mapping = aes(x=TotalMinutesAsleep, y=TotalDistance, color=TotalDistance))+
  labs(title = "Relation between distance covered by people and minutes slept",x="Total Minutes As
leep", y='Total Distance')
```



Conclusion and Recommendations

After carefully storing, formatting, manipulating, analyzing, and visualizing the data, I have come to a conclusion. The data that was provided was only for 30 individuals and of 30 days. The sample size and duration of the data is low, but two trends have been spotted very clearly.

- Users who are above the mean weight are the ones who are working out more.
- Users who are working out more are the ones who are within the range of the normal sleeping period.

Recommendations The Bellabeat app provides users with health data related to their activity, sleep, stress, menstrual cycle, and mindfulness habits. This data can help users better understand their current habits and make healthy decisions. We can use the app:

- To provide more insights about users sleeping pattern
- Find out how much calories they need to burn for a better sleep routine
- Provide a detail workout based on how calories they need to burn based on their weight
- Market our services towards people with sleeping issues

Bellabeat will win over customers' goodwill by doing this. Resolving issues that have a direct impact on people's lives will enable us to develop and grow the business. However, as we expand, we will not just make more profits, but we will also be able to solve more issues of this nature, and by helping our customers, we will provide them with a healthier way of life.

Add a new chunk by clicking the *Insert Chunk* button on the toolbar or by pressing *Ctrl+Alt+I*.

When you save the notebook, an HTML file containing the code and output will be saved alongside it (click the *Preview* button or press *Ctrl+Shift+K* to preview the HTML file).

The preview shows you a rendered HTML copy of the contents of the editor. Consequently, unlike *Knit*, *Preview* does not run any R code chunks. Instead, the output of the chunk when it was last run in the editor is displayed.