Final Project Report

Joanna Shen

2022-12-13

Introduction

Chocolate is one of the most popular candies in the world. Annual global chocolate consumption in 2022 is projected at 7.5 million tons. In the United States, chocolate accounts for 59% of all candy sales, as of 2018 and in 2021, the average spend American on chocolate was a whopping $144.90 per person. Moreover, the most recent research suggests that the average American eats approximately 3 bars of chocolate per week. From boosting our mood to making the perfect gift, chocolate plays an important role in all our lives ("31 Current Chocolate Statistics," n.d.). However, not all chocolate bars are created equal, in regard of one of their major ingredients, cocoa. How can we choose among a variety of chocolate bars when shopping at a market? These lead to the question I am interested in: What is the relationship between the cocoa percentage of a chocolate bar and its rating?

To answer this question, I will use a dataset found on Kaggle. This dataset contains expert ratings of 1795 individual chocolate bars. It focuses on plain dark chocolate bars, and the ratings, which range from 1 (lowest) to 5 (highest), reflect experiences with all the chocolate. The other columns are the name of the company manufacturing each chocolate bar, the specific regional origin of the bean used to make each bar, a reference number indicating when the review of each bar was entered in the dataset (higher = more recent), the year when the review was published, the percentage of the cocoa (darkness) of each bar, the country where the company was located, the variety of the bean (if provided), and the broad regional origin of the bean (Tatman, 2017).

Exploratory Data Analysis

The first step in our analysis is to load and organize the data. After reading in the data using the `read_csv` function, I notice that the column names contain spaces as well as some line breaks, which may cause trouble for my analysis, so I rename the columns. I also do the following data cleaning:

- Remove columns unrelated to my question of interest so that I won't use them as predictors from the data, after which the data only contains 3 columns (`review_date`, `cocoa_percent`, `rating`)
- Coerce the `review_date` column to a factor
- Remove the `%` sign from the `cocoa_percent` column and coerce it to be of base type `"double"`

To examine if the spread of ratings varies over time in order to decide whether time may be a predictor of ratings, I compute the number of ratings, average, and standard deviation for each year and store this summarized information in a table called `spread_by_year`. This table seems to show that the standard deviation decreases as time passes. This trend can be more clearly identified when I visualize it (figure 1). According to the figure, as the number of ratings recorded in each year increases, there appears a downward trend in their standard deviation, implying that reviewers are giving less extreme scores. Note that the spread of ratings in 2017 is much smaller than all the other years. This may simply be a result of the much smaller number of ratings it has, as demonstrated by both the `count` column in the `spread_by_year` table and the size of the point for 2017 in this figure.

Methodology Description and Justification

Based on my exploratory data analysis, I will first use the `lm` function to fit two linear models, one of which uses the `cocoa_percent` column as the only predictor, the other of which uses both the `cocoa_percentage` and `review_date` columns as predictors. For both of the linear models, the outcome variable will be the `rating` column. The reason I choose linear models is that they are interpretable. This advantage can help me investigate the relationship between cocoa percentage / time and rating since I can have a better idea of the magnitude of the effects of the predictors on the outcome.

Then I will split the data in training and test sets and compare two machine learning algorithms, linear regression and a classification tree. For the latter, I will use cross-validation to pick the best complexity parameter (cp). Linear regression serves as my baseline approach. The classification tree has certain advantages that make it very useful. It is highly interpretable, even more so than linear models. Besides, classification trees can model human decision processes and don't require use of dummy predictors for categorical variables. By using cross-validation, I can pick the best fitting classification tree that approximates the true relationship between the variables.

Results

Now let's dive into my analysis of whether there is a relationship between the cocoa percentage of a chocolate bar and its rating, with the year when the rating is given being an additional predictor.

Linear Models

I begin with using the `ggplot2` function `geom_smooth`, which computes and adds a regression line to plot along with confidence intervals, with the argument `method = "lm"`, which stands for linear models,to visualize the linear relationship that might exist between cocoa percentage and rating (figure 2). According to the figure, there appears to be a linear relationship, although not strong, between them. The regression line slope is negative and its intercept is close to 3.6. This figure has provided a rationale to fit a linear model for the data. By denoting the 1795 observed cocoa percentages with $x_1, \ldots, x_{1795}$, I model the 1795 ratings I am trying to predict with

$$Y_i = \beta_0 + \beta_1 x_i + \epsilon_i, i = 1, \ldots, 1795.$$

For this linear model to be useful, I have to estimate the unknown βs by finding the values that minimize the residual sum of squares (RSS). These values are called the least squares estimates (LSE) and are denoted with $\hat{\beta}_0$ and $\hat{\beta}_1$. For the data, I would write

$$RSS = \sum_{i=1}^{1795} [y_i - (\beta_0 + \beta_1 x_i)]^2$$

with y_i the observed chocolate bar's rating and x_i the observed chocolate bar's cocoa percentage. I write a function that computes the RSS for any pair of values β_0 and β_1 and plot the RSS as a function of β_1 when I keep the β_0 fixed at 3.6 (figure 3). I can see a clear minimum for β_1 at around -0.00625. However, this minimum for β_1 is for when $\beta_0 = 3.6$, a value I pick based on figure 2. I don't know if (-0.00625, 3.6) is the pair that minimizes the equation across all possible pairs. The `lm` function can help me obtain the least squares estimates by fitting the linear model.

I take a look at the model by using the `summary` function. I am given $\hat{\beta}_0 = 4.08$ and $\hat{\beta}_1 = -0.01246$, which are not far away from the (-0.00625, 3.6) pair obtained above. With an adjusted R^2 of 0.02662, it is fair to say that this isn't a very good model and it is not the most informative model, but there is a linear relationship between cocoa percentage and rating. The estimated negative slope implies that the higher the cocoa percentage of a chocolate bar is, the lower its rating will be.

Based on the previous result, in order to predict the rating of a chocolate bar, I am going to build an additional linear model that take more variables into consideration. I have already learned from the data that the spread of ratings decrease as time passes. Perhaps factoring this into consideration will help me improve my predicting power. To confirm that this is a right choice, I visualize the yearly relationships between cocoa percentage and rating (figure 4). This figure shows that there is deviation among years regarding how cocoa percentage is related to rating. However, it is quite messy given the number of years. Let's try to facet by time (figure 5). Although faceting makes it slightly more difficult to directly compare the slopes among years, it makes evaluating one at a time much more manageable and effective. Both of the figures illustrate that the slope of the regression line changes with years. Thus, I decide to fit a linear model with both the `cocoa_percent` and `review_date` columns are predictors.

I can see how adding time to the predictor affects the quality of my model, again, by using the `summary` function. This time, I am provided with much more least squares estimates given that the `review_date` column is a factor. It also shows that two of the years (2011 and 2015) can significantly impact the outcome based on a significance level $\alpha = 0.05$, but overall the effect isn't as substantial as we might expect. Note that the value of the adjusted R^2 has risen from 0.02662 to 0.04097, which means that this linear model is better than the previous one.

Machine Learning

In order to build models that can be applied to completely new datasets in the real world, I take machine learning as my next step. For simplicity, I will only use the `cocoa_percent` column as my predictor. My outcome will still be the `rating` column.

I begin with using the `createDataPartition` function included in the `caret` package to generate indexes for randomly splitting the data I have into training and test sets of equal size.

Linear regression is the first machine learning algorithm that I will try out since I have fitted two linear models in the previous section. By using the `lm` function again, I obtain an estimate of the regression line for predicting rating from cocoa percentage:

$$\hat{f}(x) = 4.03 - 0.01183x$$

where the estimated intercept and slope are close to the least squares estimates obtained above, $\hat{\beta}_0 = 4.08$ and $\hat{\beta}_1 = -0.01246$. I then use the `predict` function to predict ratings from cocoa percentages in the test set based on the fitted linear regression model. For comparison purpose, I compute the root mean squared error (RMSE) for this model.

I think viewing the outcome as categorical may improve prediction performance, so for my second machine learning algorithm, let's look at how a classification tree performs on predicting rating from cocoa percentage. I use cross-validation to choose the best complexity parameter, which is the minimum set by the algorithm for how much the RSS must improve for another partition to be added, and store all possible complexity parameter values along with their resulting RMSEs in a table called `rmse_by_cp`. This table shows that as values of complexity parameter rise, the RMSEs decrease at first, achieve their minimum of 0.47819 when the complexity parameter is equal to 0.00917, then increase. This trend can also be recognized when I visualize it (figure 6). Thus, for the data, the best fitting classification tree has complexity parameter of 0.00917. I then use the `predict` function again to predict ratings from cocoa percentages in the test set based on this best fitting classification tree. Since the predicted ratings have fewer levels than the observed ratings, I choose not to use the `confusionMatrix` function to obtain the accuracy. To compare this algorithm with linear regression, I compute its RMSE.

I store the two computed RMSEs in a table called `rmse_by_model`. From this table, it is clear that the RMSE of the classification tree is smaller than the RMSE of linear regression ($0.44906 < 0.46059$), which means that the classification tree is better than linear regression. Therefore, I choose to move forward with the classification tree.

I am going to end this section with a plot of the estimated ratings resulting from the classification tree (figure 7). According to the figure, the classification tree is able to approximately estimate the trend of ratings, but its accuracy of predicting rating given specific cocoa percentage may be quite low.

Conclusion

In this project, I explored the following question: What is the relationship between the cocoa percentage of a chocolate bar and its rating? By fitting linear models, building machine learning algorithms, which include linear regression and a classification tree, and using cross-validation, and comparing different models and algorithms, I obtained the following results:

- There isn't a strong relationship between cocoa percentage and rating
- Using time in addition to cocoa percentage as predictors can help predict rating
- Classification tree performs better than linear regression

I think my analysis was partly successful and partly not successful. On one hand, it could be considered successful because my predicting power has been improved along the way and I was able to find a final model, which is the classification tree. On the other hand, it could be considered as not successful because the accuracy is low.

If I had more time, I would take more potential predictors available in the dataset into account, such as the country where the company manufacturing each chocolate bar was located and the broad regional origin of the bean used to make each bar. I would also try other machine learning algorithms, such as random forests, to see whether the RMSE can decrease and the accuracy can increase. The reason I would consider building random forests as a possible direction for future work is that random forests address the shortcomings of classification trees. By averaging multiple classification trees (a forest of trees constructed with randomness), random forests can improve prediction performance and reduce instability.

References

31 current chocolate statistics (chocolate market data 2022). (n.d.). Retrieved December 14, 2022, from https://damecacao.com/chocolate-statistics/

Tatman, R. (2017). Chocolate bar ratings [Data set]. Kaggle. https://www.kaggle.com/datasets/rtatman/chocolate-bar-ratings

Appendix

```
library(tidyverse)
library(dplyr)
library(stringr)
library(ggplot2)
library(caret)
```

```
d <- read_csv("flavors_of_cacao.csv", show_col_types = FALSE)
head(d)
```

```
## # A tibble: 6 x 9
##   Company \n(Make~1 Speci~2   REF Revie~3 Cocoa~4 Compa~5 Rating Bean\~6 Broad~7
##   <chr>             <chr>   <dbl>   <dbl> <chr>   <chr>    <dbl> <chr>   <chr>
## 1 A. Morin          Agua G~  1876    2016 63%     France    3.75         Sao To~
## 2 A. Morin          Kpime    1676    2015 70%     France    2.75         Togo
## 3 A. Morin          Atsane   1676    2015 70%     France    3            Togo
## 4 A. Morin          Akata    1680    2015 70%     France    3.5          Togo
## 5 A. Morin          Quilla   1704    2015 70%     France    3.5          Peru
## 6 A. Morin          Carene~  1315    2014 70%     France    2.75 Criollo Venezu~
## # ... with abbreviated variable names 1: 'Company \n(Maker-if known)',
## #   2: 'Specific Bean Origin\nor Bar Name', 3: 'Review\nDate',
## #   4: 'Cocoa\nPercent', 5: 'Company\nLocation', 6: 'Bean\nType',
## #   7: 'Broad Bean\nOrigin'
```

```
colnames(d)
```

```
## [1] "Company \n(Maker-if known)"        "Specific Bean Origin\nor Bar Name"
## [3] "REF"                               "Review\nDate"
## [5] "Cocoa\nPercent"                    "Company\nLocation"
## [7] "Rating"                            "Bean\nType"
## [9] "Broad Bean\nOrigin"
```

```
colnames(d) <- c("company", "specific_bean_origin", "ref", "review_date", "cocoa_percent",
                 "company_location", "rating", "bean_type", "broad_bean_origin")
d <- d[,c(4, 5, 7)]
class(d$review_date)
```

```
## [1] "numeric"
```

```
d$review_date <- as.factor(d$review_date)
class(d$cocoa_percent)
```

```
## [1] "character"
```

```
d$cocoa_percent <- str_replace_all(d$cocoa_percent, "%", "")
d$cocoa_percent <- as.numeric(d$cocoa_percent)

head(d)
```

```
## # A tibble: 6 x 3
##   review_date cocoa_percent rating
##   <fct>               <dbl>  <dbl>
## 1 2016                   63   3.75
## 2 2015                   70   2.75
## 3 2015                   70   3
## 4 2015                   70   3.5
## 5 2015                   70   3.5
## 6 2014                   70   2.75
```

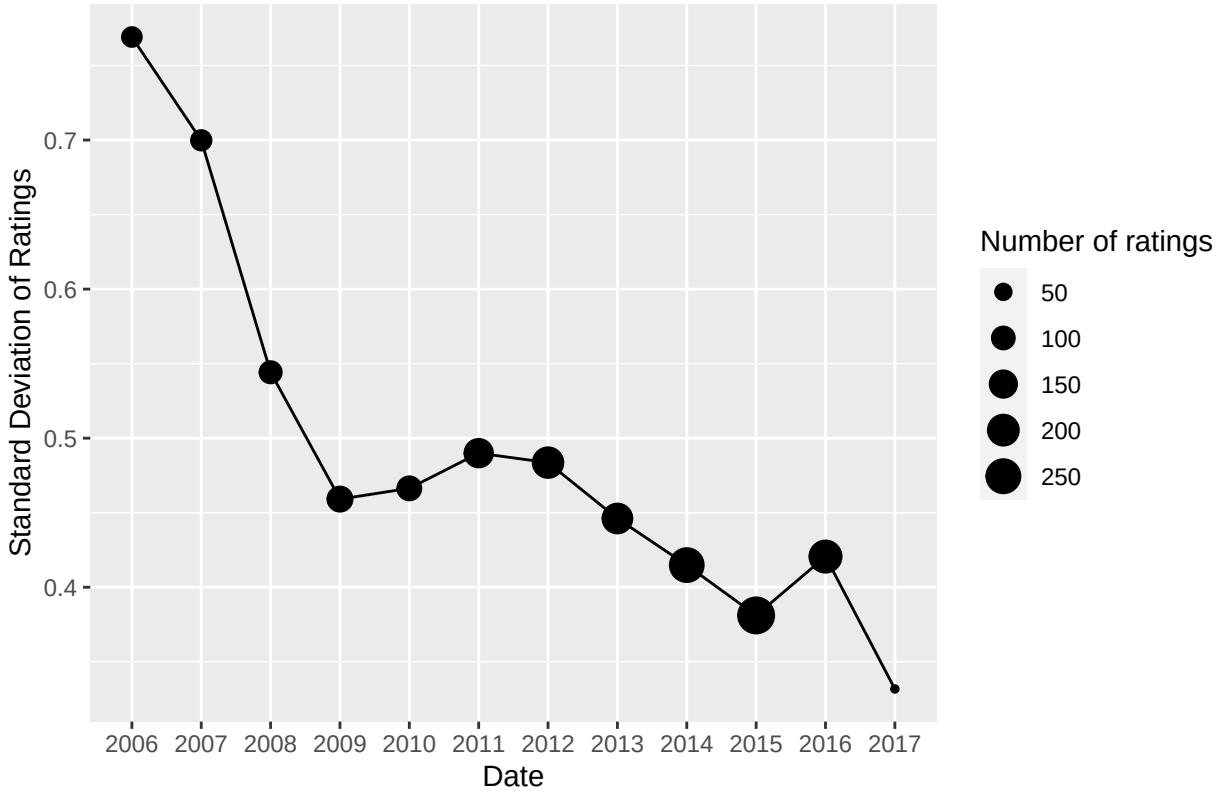
```r
spread_by_year <- d %>%
  group_by(review_date) %>%
  summarize(count = n(), avg = mean(rating), sd = sd(rating))

spread_by_year
```

```
## # A tibble: 12 x 4
##    review_date count   avg    sd
##    <fct>       <int> <dbl> <dbl>
##  1 2006           72  3.12 0.769
##  2 2007           77  3.16 0.700
##  3 2008           93  2.99 0.544
##  4 2009          123  3.07 0.459
##  5 2010          111  3.15 0.466
##  6 2011          165  3.26 0.490
##  7 2012          195  3.18 0.484
##  8 2013          184  3.20 0.446
##  9 2014          247  3.19 0.415
## 10 2015          285  3.25 0.381
## 11 2016          219  3.23 0.421
## 12 2017           24  3.31 0.332
```

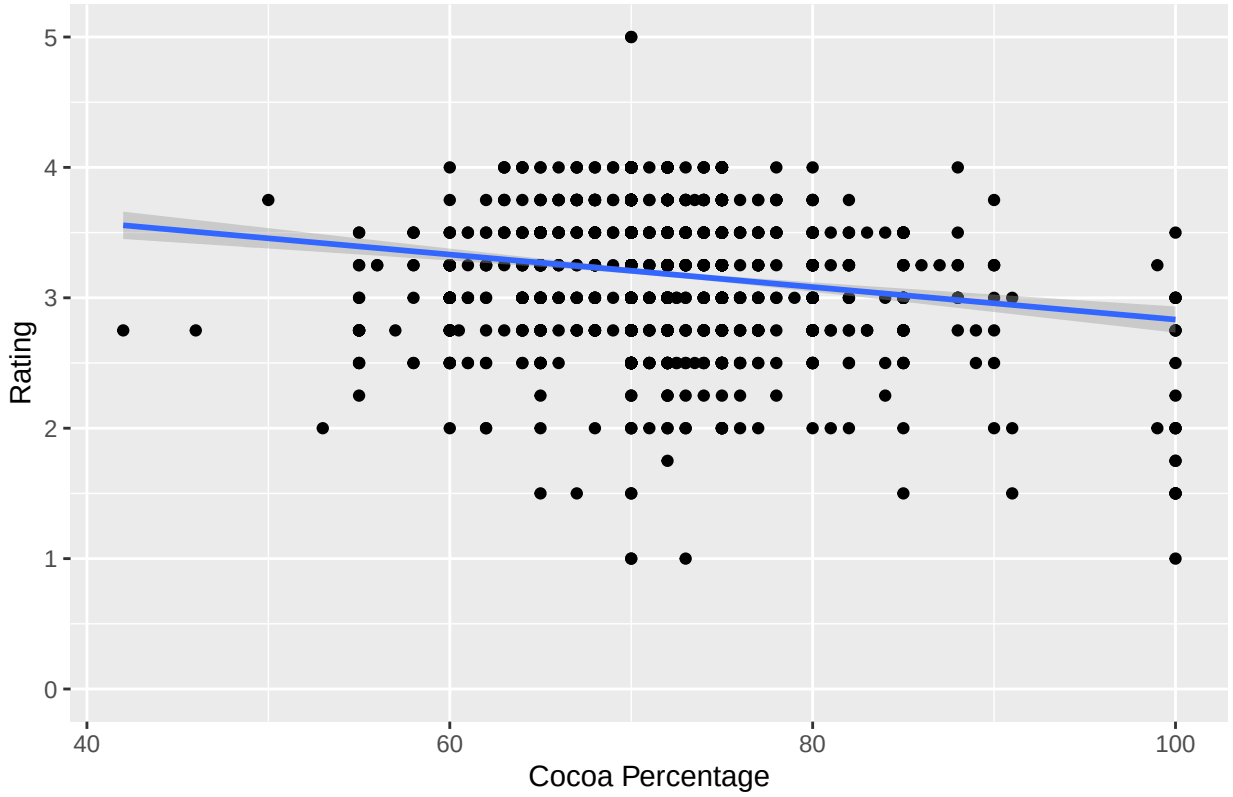
```r
spread_by_year %>%
  ggplot(aes(x = review_date, y = sd, group = 1)) +
  geom_point(aes(size = count)) + # change point size based on number of ratings
  geom_line(size = 0.5) +
  scale_size(breaks = seq(0, 250, 50), name = "Number of ratings") +
  labs(x = "Date",
       y = "Standard Deviation of Ratings",
       title = "Figure 1: Standard deviation of ratings over time")
```

Figure 1: Standard deviation of ratings over time




```
d %>%
  ggplot(aes(x = cocoa_percent, y = rating)) +
  geom_point() +
  geom_smooth(method = "lm", formula = y ~ x) +
  coord_cartesian(ylim = c(0, 5)) +
  ggtitle("Figure 2: Rating vs. cocoa percentage") +
  xlab("Cocoa Percentage") +
  ylab("Rating")
```

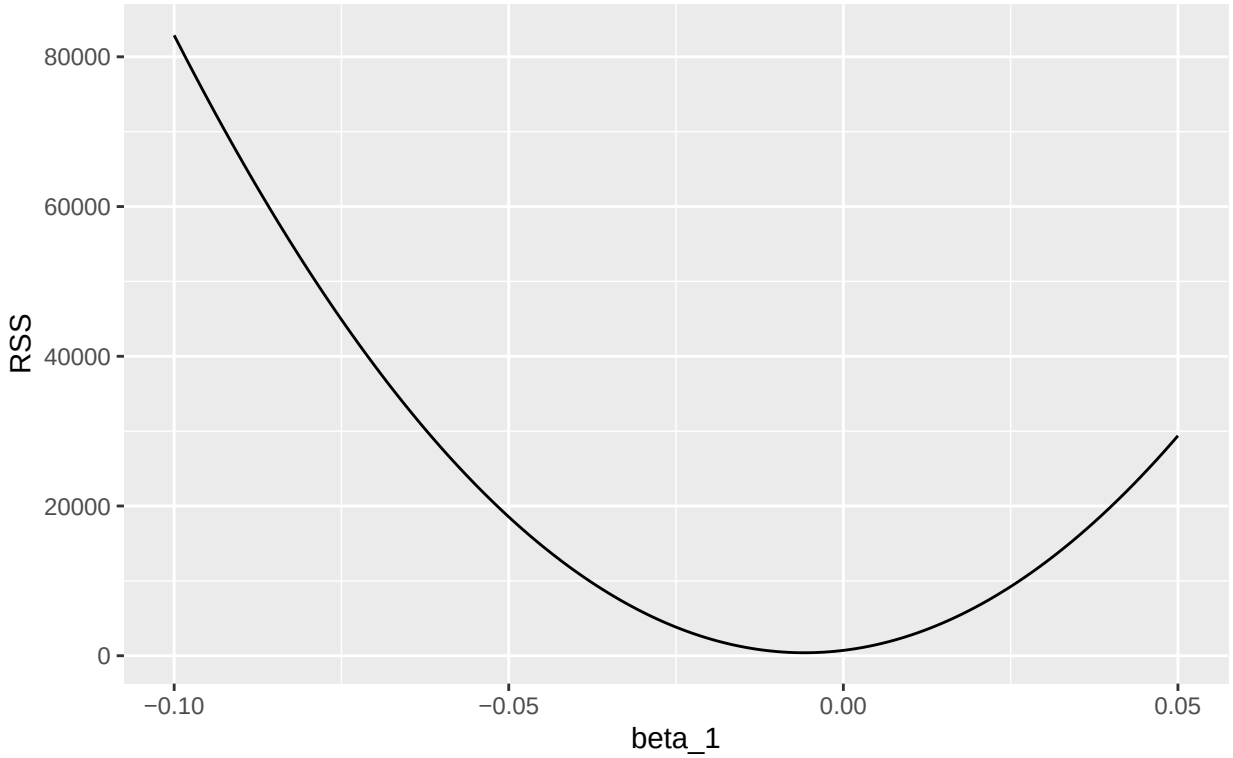
Figure 2: Rating vs. cocoa percentage




```
rss <- function(beta0, beta1, data){
  resid <- d$rating - (beta0 + beta1 * d$cocoa_percent)
  return(sum(resid^2))
}
```

```
beta1 <- seq(-0.1, 0.05, len = nrow(d))
results <- data.frame(beta1 = beta1,
                      rss = sapply(beta1, rss, beta0 = 3.6))
results %>%
  ggplot(aes(x = beta1, y = rss)) +
  geom_line() +
  ggtitle("Figure 3: Residual sum of squares (RSS) as a function of beta_1\nwith fixed beta_0") +
  xlab("beta_1") +
  ylab("RSS")
```

Figure 3: Residual sum of squares (RSS) as a function of beta_1 with fixed beta_0



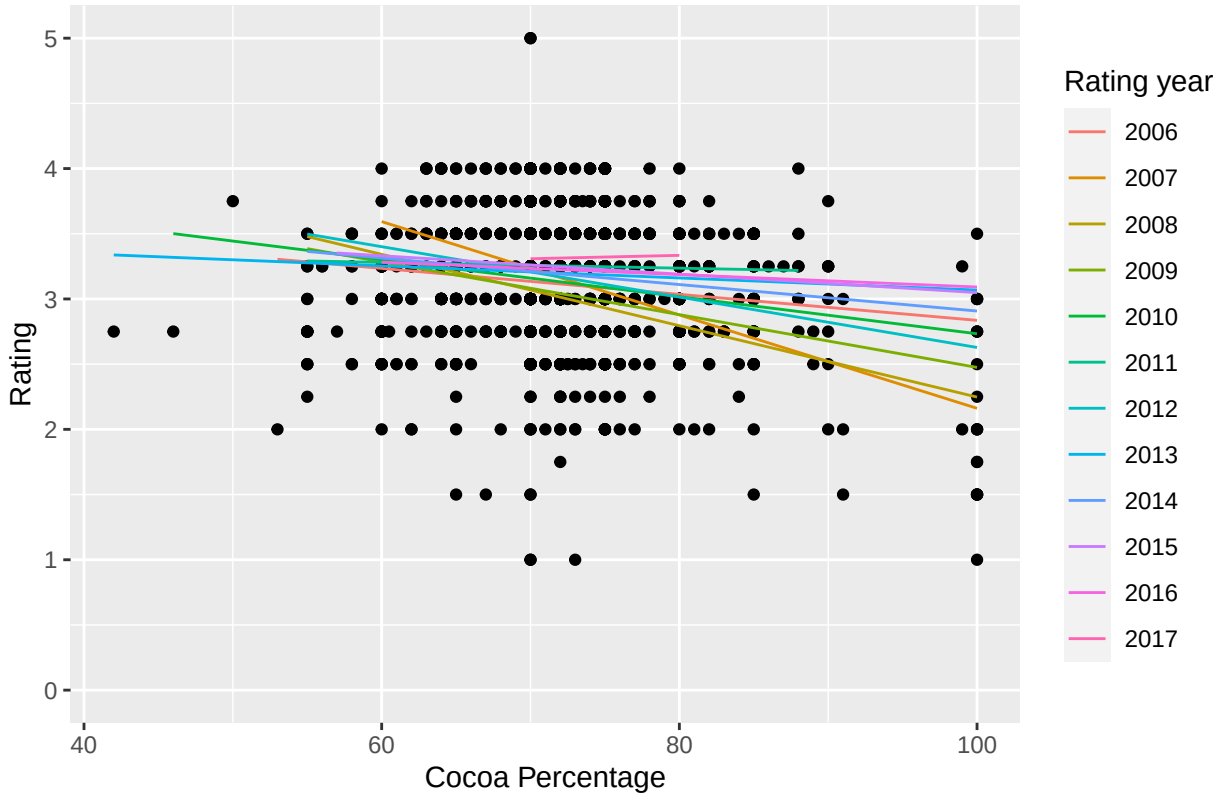

```
fit_1 <- lm(rating ~ cocoa_percent, data = d)
summary(fit_1)
```

```
##
## Call:
## lm(formula = rating ~ cocoa_percent, data = d)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.2071 -0.3196  0.0429  0.3178  1.7929
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)    4.079388   0.126757  32.183  < 2e-16 ***
## cocoa_percent -0.012461   0.001761  -7.076 2.12e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.4717 on 1793 degrees of freedom
## Multiple R-squared:  0.02717,    Adjusted R-squared:  0.02662
## F-statistic: 50.07 on 1 and 1793 DF,  p-value: 2.122e-12
```

```
d %>%
  ggplot(aes(x = cocoa_percent, y = rating, group = review_date)) +
  geom_point() +
  geom_smooth(aes(color = review_date),
```

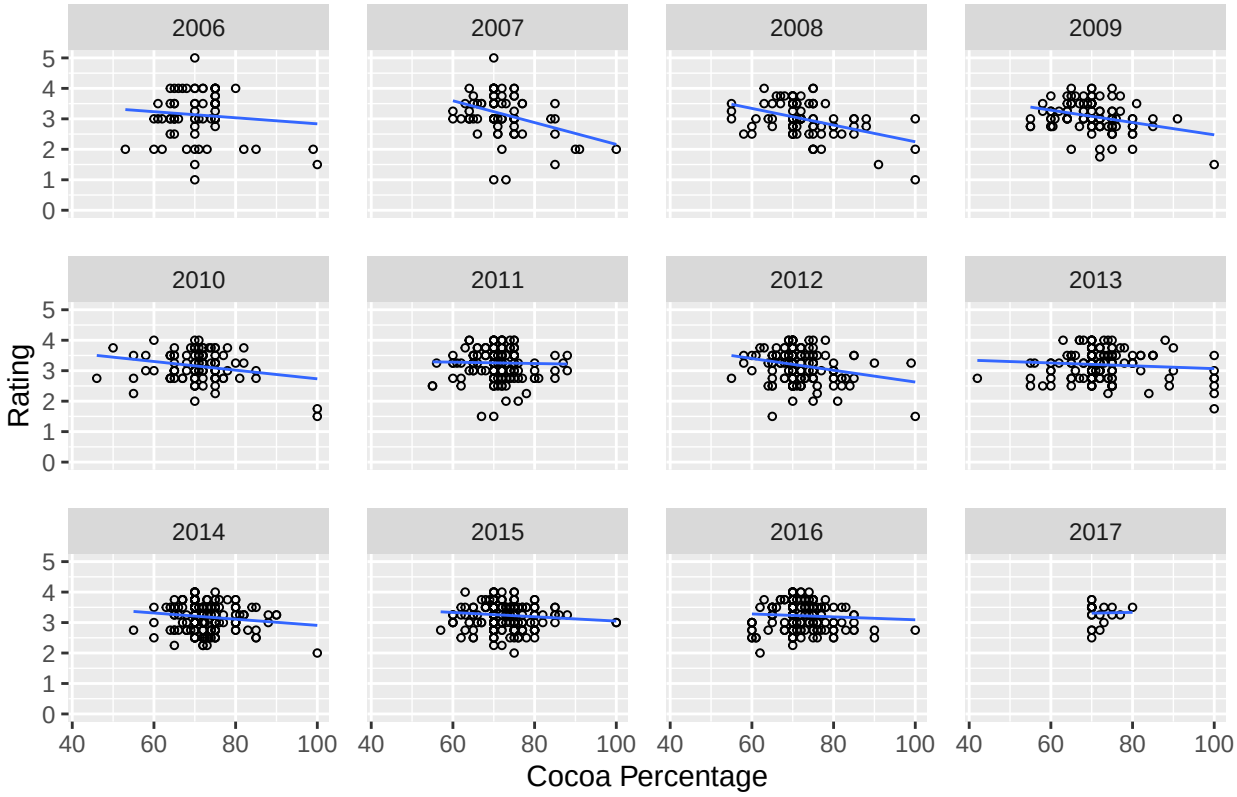
```
            method = "lm", formula = y ~ x, se = FALSE, size = 0.5) +
  coord_cartesian(ylim = c(0, 5)) +
  ggtitle("Figure 4: Rating vs. cocoa percentage with yearly relationships") +
  xlab("Cocoa Percentage") +
  ylab("Rating") +
  scale_color_discrete(name = "Rating year")
```

Figure 4: Rating vs. cocoa percentage with yearly relationships




```
d %>%
  ggplot(aes(x = cocoa_percent, y = rating, group = review_date)) +
  geom_point(shape = 21, size = 1) +
  geom_smooth(method = "lm", formula = y ~ x, se = FALSE, size = 0.5) +
  coord_cartesian(ylim = c(0, 5)) +
  ggtitle("Figure 5: Rating vs. cocoa percentage by years") +
  xlab("Cocoa Percentage") +
  ylab("Rating") +
  facet_wrap(vars(review_date)) +
  theme(panel.spacing = unit(1, "lines"))
```

Figure 5: Rating vs. cocoa percentage by years




```
fit_2 <- lm(rating ~ cocoa_percent + review_date, data = d)
summary(fit_2)
```

```
##
## Call:
## lm(formula = rating ~ cocoa_percent + review_date, data = d)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.18813 -0.27847  0.02432  0.30320  1.86235
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)       4.023050   0.136368  29.501  < 2e-16 ***
## cocoa_percent    -0.012649   0.001756  -7.201 8.76e-13 ***
## review_date2007   0.050479   0.076772   0.658   0.5109
## review_date2008  -0.108887   0.073552  -1.480   0.1389
## review_date2009  -0.058873   0.069477  -0.847   0.3969
## review_date2010   0.020857   0.070844   0.294   0.7685
## review_date2011   0.130677   0.066125   1.976   0.0483 *
## review_date2012   0.059886   0.064568   0.927   0.3538
## review_date2013   0.088028   0.065118   1.352   0.1766
## review_date2014   0.080120   0.062741   1.277   0.2018
## review_date2015   0.134317   0.061777   2.174   0.0298 *
## review_date2016   0.110615   0.063614   1.739   0.0822 .
## review_date2017   0.194351   0.110352   1.761   0.0784 .
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.4682 on 1782 degrees of freedom
## Multiple R-squared:  0.04739,    Adjusted R-squared:  0.04097
## F-statistic: 7.387 on 12 and 1782 DF,  p-value: 2.096e-13
```

```
set.seed(1994)
d <- d[,-1]
y <- d$rating
test_index <- createDataPartition(y, times = 1, p = 0.5, list = FALSE)

train_set <- d %>% slice(-test_index)
test_set <- d %>% slice(test_index)
```

```
train_lm <- lm(rating ~ cocoa_percent, data = train_set)
train_lm$coef
```

```
##   (Intercept) cocoa_percent
##    4.02820494   -0.01182989
```

```
y_hat_lm <- predict(train_lm, test_set)
rmse_lm <- sqrt(mean((y_hat_lm - test_set$rating)^2))
```

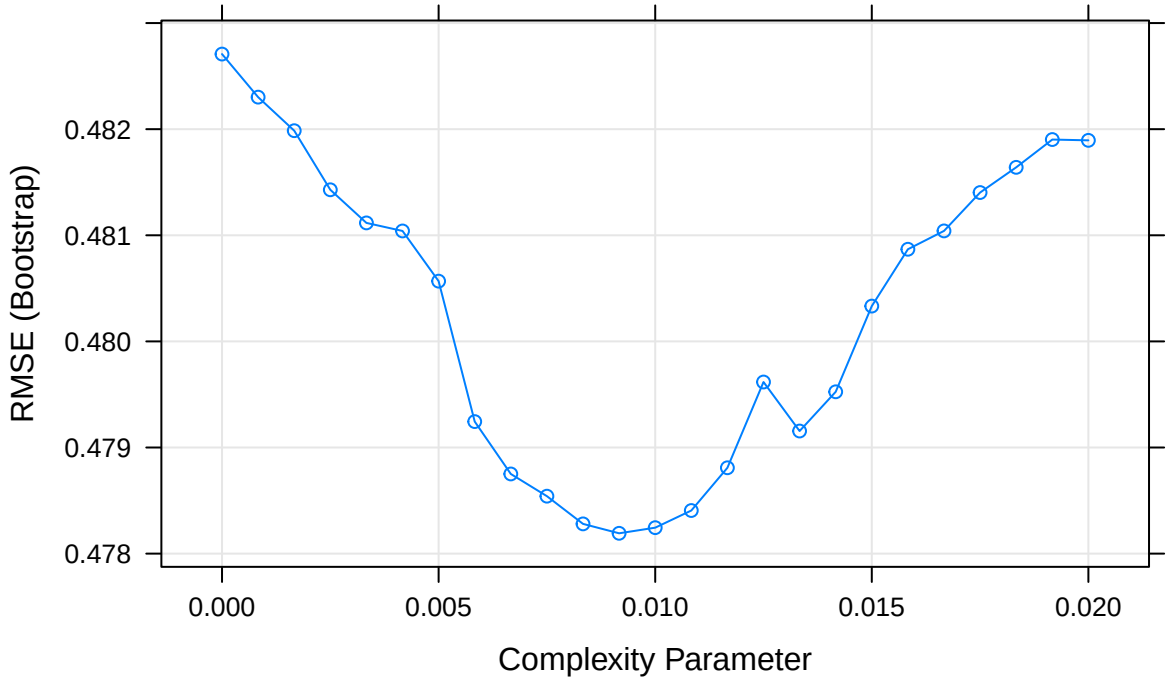
```
train_rpart <- train(rating ~ cocoa_percent,
                     method = "rpart",
                     tuneGrid = data.frame(cp = seq(0.0, 0.02, len = 25)),
                     data = train_set)
rmse_by_cp <- data.frame(cp = train_rpart$results$cp,
                         rmse = train_rpart$results$RMSE)
rmse_by_cp
```

```
##              cp      rmse
## 1  0.0000000000 0.4827077
## 2  0.0008333333 0.4823017
## 3  0.0016666667 0.4819857
## 4  0.0025000000 0.4814287
## 5  0.0033333333 0.4811166
## 6  0.0041666667 0.4810406
## 7  0.0050000000 0.4805677
## 8  0.0058333333 0.4792444
## 9  0.0066666667 0.4787506
## 10 0.0075000000 0.4785414
## 11 0.0083333333 0.4782804
## 12 0.0091666667 0.4781916
## 13 0.0100000000 0.4782445
## 14 0.0108333333 0.4784063
## 15 0.0116666667 0.4788090
## 16 0.0125000000 0.4796172
## 17 0.0133333333 0.4791552
## 18 0.0141666667 0.4795250
## 19 0.0150000000 0.4803329
```

```
## 20 0.0158333333 0.4808683
## 21 0.0166666667 0.4810413
## 22 0.0175000000 0.4814029
## 23 0.0183333333 0.4816403
## 24 0.0191666667 0.4819023
## 25 0.0200000000 0.4818947
```

```
plot(train_rpart, main = "Figure 6: Root mean squared error (RMSE)\nwith different values of complexity
```

Figure 6: Root mean squared error (RMSE)
with different values of complexity parameter (cp)




```
y_hat_rpart <- predict(train_rpart, test_set)
table(y_hat_rpart)
```

```
## y_hat_rpart
## 2.48611111111111 2.99038461538462 3.09216589861751             3.25
##               17               45              227              609
```

```
with(test_set, table(rating))
```

```
## rating
##    1  1.5 1.75    2 2.25  2.5 2.75    3 3.25  3.5 3.75    4    5
##    1    7    1    9    8   64  135  169  153  196  107   47    1
```

```
rmse_rpart <- sqrt(mean((y_hat_rpart - test_set$rating)^2))
```

```
rmse_by_model <- data.frame(model = c("Linear regression", "Classification tree"),
                            rmse = c(rmse_lm, rmse_rpart))
rmse_by_model
```

```
##                 model      rmse
## 1   Linear regression 0.4605912
## 2 Classification tree 0.4490550
```

```
test_set %>%
  mutate(y_hat_rpart = y_hat_rpart) %>%
  ggplot() +
  geom_point(aes(x = cocoa_percent, y = rating)) +
  geom_step(aes(x = cocoa_percent, y = y_hat_rpart), col = "red") +
  ggtitle("Figure 7: Rating vs. cocoa percentage with prediction by classification tree") +
  xlab("Cocoa Percentage") +
  ylab("Rating")
```

Figure 7: Rating vs. cocoa percentage with prediction by classification tree