CYO - Chocolate Bars Recommendation System

Sejal Arora

14/05/2021

Introduction

Chocolate has proved to be a popular food product consumed by millions every day. A major reason behind it is its unique, rich and sweet taste. According to the World Cocoa Foundation, people all around the world consume more than three million tons of cocoa beans every year. However, tastes of people all around the world differ with respect to the region they are living in. Moreover, there are various factors that contribute to the popularity of a particular chocolate bar since not all chocolate bars are similar. The name of the chocolate bar, the name of the company that manufactured the chocolate bar, its country, percentage of cocoa present, the type of cocoa bean used, the origin of the cocoa bean, etc. are the various variables that affect the popularity of chocolate bars. As we move ahead with this project, we will come across these variables and deeply study their overall impact on the Chocolate Rating System and also on the chocolate bars.

The dataset used in this project has been taken from Kaggle (https://www.kaggle.com/rtatman/chocolate-bar-ratings). It consists of expert ratings of over 1700 chocolate bars all around the globe along with their specific information.

It is based on Flavors of Cacao Rating System (http://flavorsofcacao.com/index.html):

- 5= Elite (Transcending beyond the ordinary limits)
- 4= Premium (Superior flavor development, character and style)
- 3= Satisfactory(3.0) to praiseworthy(3.75) (well made with special qualities)
- 2= Disappointing (Passable but contains at least one significant flaw)
- 1= Unpleasant (mostly unpalatable)

The goal of this project is to predict the Chocolate Bar Rating Class using the Chocolate Bar Ratings given in the dataset by using multiple machine learning methods to achieve a higher accuracy of the prediction. For that, we first download the dataset, standardize and customize it on the basis of International Organization for Standardization (ISO) - 3166 codes and further explore the dataset. After that, we will partition the dataset into Training and Validation set and use various techniques/methods to achieve the best accuracy on the validation set.

This report doesn't contain the entire code. All the 3 scripts can be found at my GitHub respository.

Dataset Generation and Analysis

Downloading the dataset and modifying

```
# We then download the csv file from the git link
link_datasetchocolate <- "https://raw.githubusercontent.com/sejalarora21/CYO-project-/main/read_csv.csv

# Next, we read file into raw table and remove non-printable characters
datachocolate <- read.csv(gsub("[^[:print:]]","",link_datasetchocolate),
                          na = c(""," ","NA"))

# Using rworldmap package, we create a Country-Region mapping
data_countryregion <- countryRegions %>% mutate(CountryName = ADMIN,
CountryCode = ISO3,GeoRegion = GEO3) %>% filter(!is.na(GeoRegion)) %>%
select(CountryName, CountryCode, GeoRegion)
```

Firstly, we download the required dataset from a csv file via the link “https://raw.githubusercontent.com/sejalarora21/CYO-project-/main/read_csv.csv”. Next, we remove the non-printable characters present in the file for smooth analysis.

Initital Exploration

```
#####################################
# ORIGINAL DATA - INITIAL EXPLORATION
#####################################

# structure of the dataset
str(datachocolate)
```

```
## 'data.frame':    1795 obs. of  9 variables:
##  $ Company...Maker.if.known.       : chr  "A. Morin" "A. Morin" "A. Morin" "A. Morin" ...
##  $ Specific.Bean.Origin.or.Bar.Name: chr  "Agua Grande" "Kpime" "Atsane" "Akata" ...
##  $ REF                             : int  1876 1676 1676 1680 1704 1315 1315 1315 1319 1319 ...
##  $ Review.Date                     : int  2016 2015 2015 2015 2015 2014 2014 2014 2014 2014 ...
##  $ Cocoa.Percent                   : chr  "63%" "70%" "70%" "70%" ...
##  $ Company.Location                : chr  "France" "France" "France" "France" ...
##  $ Rating                          : num  3.75 2.75 3 3.5 3.5 2.75 3.5 3.5 3.75 4 ...
##  $ Bean.Type                       : chr  NA NA NA NA ...
##  $ Broad.Bean.Origin               : chr  "Sao Tome" "Togo" "Togo" "Togo" ...
```

```
# columns in the dataset
ncol(datachocolate)
```

```
## [1] 9
```

```
# rows in the dataset
nrow(datachocolate)
```

```
## [1] 1795
```

```
# missing vales (if any) in the dataset
sum(is.na(datachocolate))
```

```
## [1] 962
```

```
# renaming the columns
name_of_columns <- c("CompanyName", "ChocolateBarName", "Reference", "ReviewYear",
```

```r
                       "CocoaPercentage", "CompanyCountry", "Rating",
                       "BeanType", "BeanOrigin")

names(datachocolate) <-name_of_columns

# missing values (if any) in each column of the dataset
missingvalues <- tibble("Column Name" = c("CompanyName", "ChocolateBarName",
                                          "Reference", "ReviewYear", "CocoaPercentage",
                                          "CompanyCountry", "Rating", "BeanType", "BeanOrigin"),
                        "Missing Values" = c(sum(is.na(datachocolate$CompanyName)),
                                             sum(is.na(datachocolate$ChocolateBarName)),
                                             sum(is.na(datachocolate$Reference)),
                                             sum(is.na(datachocolate$ReviewYear)),
                                             sum(is.na(datachocolate$CocoaPercentage)),
                                             sum(is.na(datachocolate$CompanyCountry)),
                                             sum(is.na(datachocolate$Rating)),
                                             sum(is.na(datachocolate$BeanType)),
                                             sum(is.na(datachocolate$BeanOrigin))))
```

The dataset consists of ***1795*** rows, ***9*** columns and ***962*** missing values.

```r
# missing Value counts
missingvalues %>%
kable() %>%
kable_styling(bootstrap_options = ("bordered"), full_width = FALSE,
              font_size = 15, position = "center", latex_options = "hold_position")
```

Column Name	Missing Values
CompanyName	0
ChocolateBarName	0
Reference	0
ReviewYear	0
CocoaPercentage	0
CompanyCountry	0
Rating	0
BeanType	888
BeanOrigin	74

The variables present in our dataset are as follows:

- `CompanyName`: Name of the company manufacturing the Chocolate bar.
- `ChocolateBarName`: The name of the chocolate bar, its species, or its geo-region of origin.
- `Reference`: A numeric value of the chocolate bar when the review was published.A higher value refers to a recent review.
- `ReviewYear`: The year when the review of the chocolate bar was published.
- `CocoaPercentage`: Percentage of Cocoa in the particular chocolate bar.
- `CompanyCountry`: Name of the country where the manufacturing company is present.
- `Rating`: Expert rating for the Chocolate bar from 1 to 5 with 0.25 increment.
- `BeanType`: The species/breed of bean used. It may or may not be provided.
- `BeanOrigin`: The Geo-region of origin/ Country of the bean. It may or may not be provided.

Here is an example containing first 6 rows of the data set.

```r
# first 6 rows of the dataset
head(datachocolate) %>%
kable() %>%
kable_styling(bootstrap_options = ("bordered"), full_width = FALSE,
              font_size = 6, position = "center", latex_options = "hold_position")
```

CompanyName	ChocolateBarName	Reference	ReviewYear	CocoaPercentage	CompanyCountry	Rating	BeanType	BeanOrigin
A. Morin	Agua Grande	1876	2016	63%	France	3.75	NA	Sao Tome
A. Morin	Kpime	1676	2015	70%	France	2.75	NA	Togo
A. Morin	Atsane	1676	2015	70%	France	3.00	NA	Togo
A. Morin	Akata	1680	2015	70%	France	3.50	NA	Togo
A. Morin	Quilla	1704	2015	70%	France	3.50	NA	Peru
A. Morin	Carenero	1315	2014	70%	France	2.75	Criollo	Venezuela

Customize and Standardize the dataset - Preprocessing

Since the data is not normalized, we will transform it so that the data is easier to explore further.

We apply the following cleaning and transformation rules to our dataset:

- Based on Country Name or Sub-Region of the Country according to the International Organization for Standardization (ISO) -3166 codes, we will first standardize the `BeanOrigin` column data.
- Again, based on ISO 3166 codes, we will correct the misspelled countries in `CompanyCountry` column.
- 5 main type of bean groups are: Criollo, Forastero, Nacional, Trinitario and Blend. We will group `BeanType` column on the basis of these groups.
- Next, if

(i) there are missing values (NA) for multiple countries under the 'BeanOrigin' and 'BeanType' column;

(ii) there is 'Blend' or 'blend' or ',' in the 'ChocolateBarName' column and if there are missing values (NA) in the 'BeanType' column,

then we replace missing value for 'BeanType' column to 'Blend' column. - We next convert the 'CocoaPercentage' column to Numeric by removing the % sign and rounding it to nearest integer

Moreover, we create new variables that can be useful to build our preditive model:

- We create a new variable `RatingScale` on the basis of ***Flavor Of Cocoa Rating System***.
- We create a new variable `BeanOriginGeoRegion` by including the Geo-region based on the `BeanOrigin` column.
- We create a new variable `CompanyGeoRegion` by including the Geo-region based on the `CompanyCountry` column.

The `BeanOrigin` consists of pipe-separated values and this variable holds importance when it comes to predicting the Chocolate bar ratings. Hence, we extract individual values only.

Next, we convert the `Rating` column to numeric and all other columns to factor data.

After preprocessing the data, this is how the required `cleandata` data looks like:

```
# cleandata example
head(cleandata) %>%
kable() %>%
kable_styling(bootstrap_options = ("bordered"), full_width = FALSE,
              font_size = 3, position = "center", latex_options = "hold_position")
```

ChocolateBarName	CompanyName	CompanyCountry	CompanyGeoRegion	BeanType	BeanOrigin	BeanOriginGeoRegion	CocoaPercentage	ReviewYear	Rating	RatingClass
Agua Grande	A. Morin	France	Western Europe	NA	Sao Tome and Principe	Central Africa	63	2016	3.75	30-Satisfactory
Kpime	A. Morin	France	Western Europe	NA	Togo	Western Africa	70	2015	2.75	20-Disappointing
Atsane	A. Morin	France	Western Europe	NA	Togo	Western Africa	70	2015	3.00	30-Satisfactory
Akata	A. Morin	France	Western Europe	NA	Togo	Western Africa	70	2015	3.50	30-Satisfactory
Quilla	A. Morin	France	Western Europe	NA	Peru	South America	70	2015	3.50	30-Satisfactory
Carenero	A. Morin	France	Western Europe	Criollo	Venezuela	South America	70	2014	2.75	20-Disappointing

Hence, finally, the variables present in the final dataset after preprocessing, customizing and standardizing are as follows:

```
# missing values
dataset %>%
kable() %>%
kable_styling(bootstrap_options = ("bordered"), full_width = FALSE,
              font_size = 15, position = "center", latex_options = "hold_position")
```

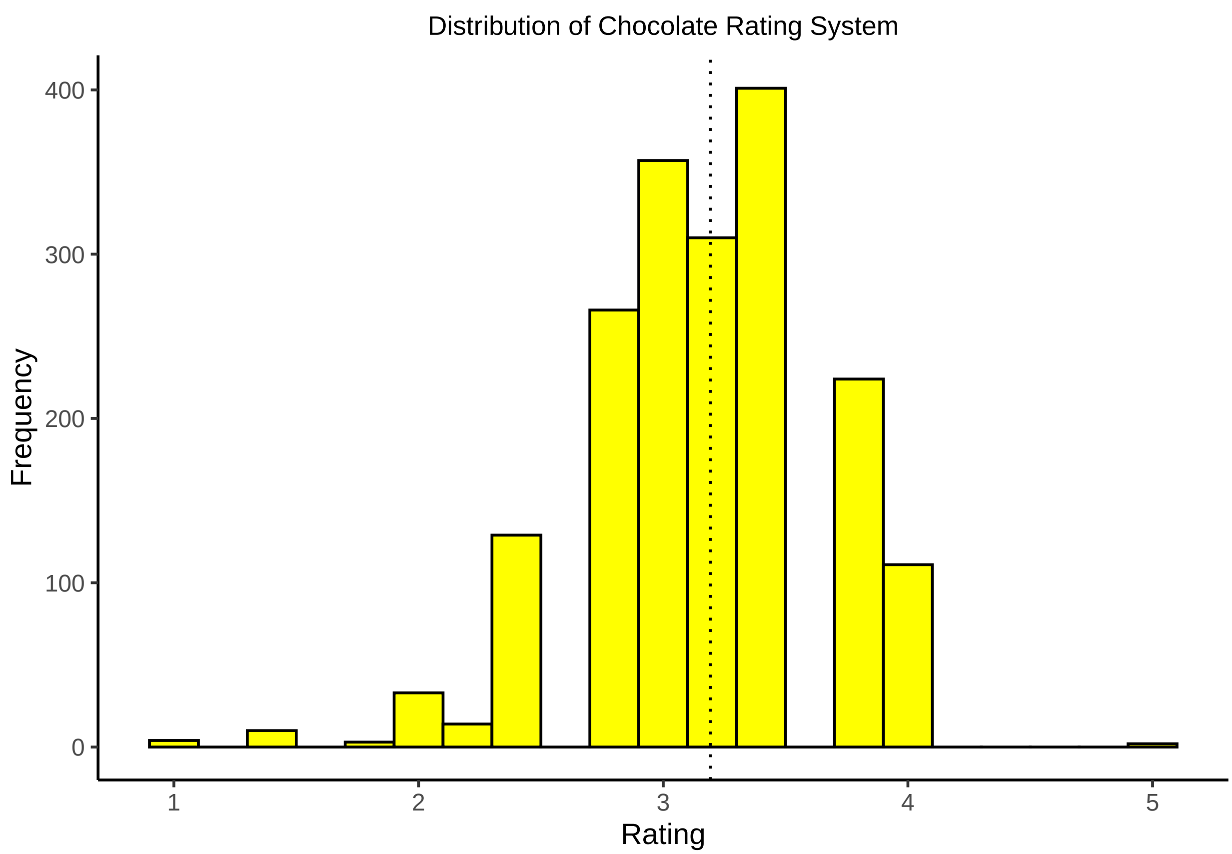
Feature Name	Data Type	Distinct Values	Missing Values
ChocolateBarName	character	1039	0
CompanyName	factor	416	0
CompanyCountry	factor	54	0
CompanyGeoRegion	factor	17	0
BeanType	factor	6	607
BeanOrigin	factor	54	74
BeanOriginGeoRegion	factor	13	74
CocoaPercentage	numeric	42	0
ReviewYear	factor	12	0
Rating	numeric	13	0
RatingClass	factor	5	0

Data Exploration

After customizing the dataset for better analysation and visualization, we first explore the `cleandata` dataset using some visualization techniques to get a deeper understanding of the data so that we can apply appropriate methods ahead.

Distribution of overall Chocolate Bar Ratings

```
################################################
# EXPLORATIOON
################################################

# Distribution of overall Chocolate Bar Ratings
ratings <- mean(cleandata$Rating)

cleandata %>%
ggplot(aes(Rating)) +
geom_histogram(color = "black", fill = "yellow", binwidth = 0.2) +
geom_vline(xintercept = ratings, col = "black",linetype = "dotted") +
labs(title = "Distribution of Chocolate Rating System", x = "Rating", y = "Frequency")+
theme_classic()+
theme(plot.title = element_text(size = 10, color = "black", hjust = 0.5))
```

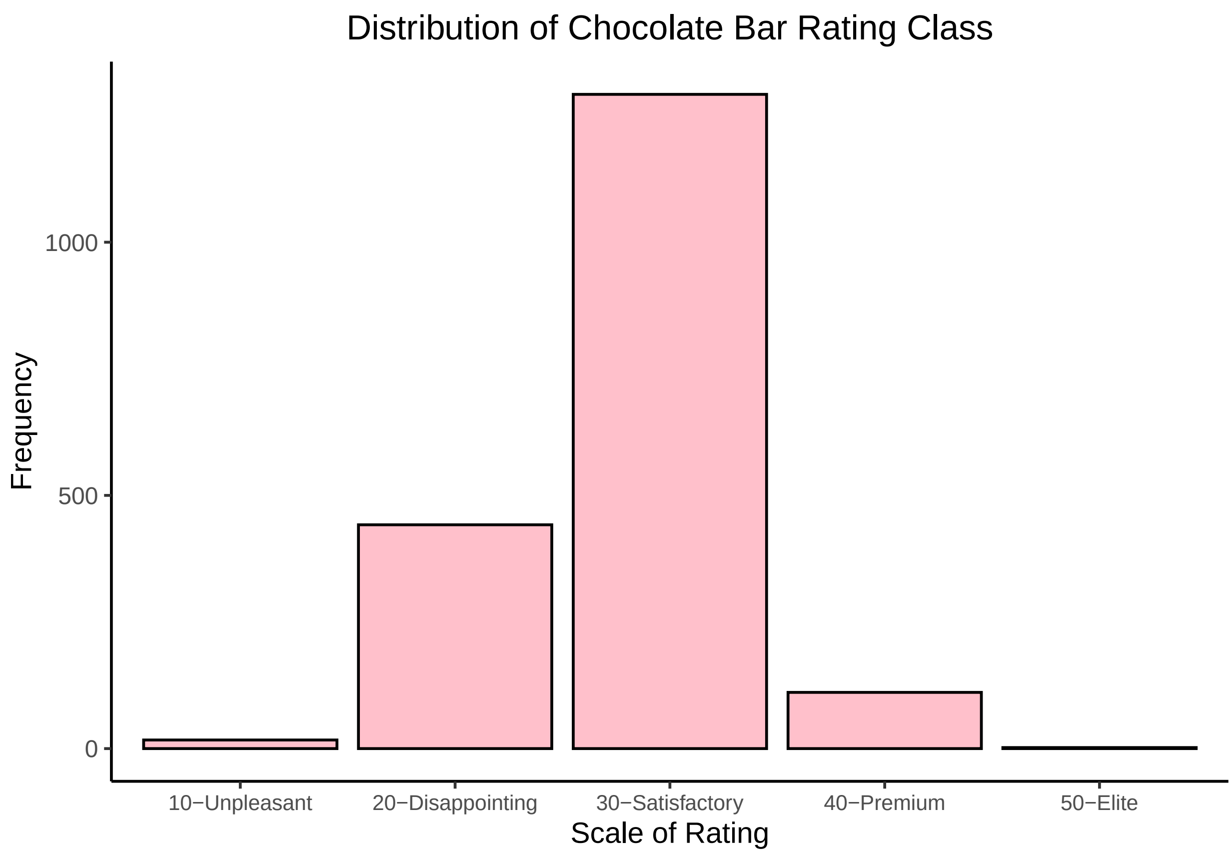


The rating ditribution of the dataset is depicted in the figure *"Distribution of Chocolate Rating System"*. The vertical dashed line depicts the overall rating average μ (***3.192999***) across all chocolate bars. We notice also that the rating range is from 1 to 4 with an exception for a few Chocolate bars that are rated as Elite or Unpleasant chocolate.

The figure also depicts that the most common rating is 3.5 (400 ratings) followed by 3.0.

Distribution of Chocolate Bar Rating Class

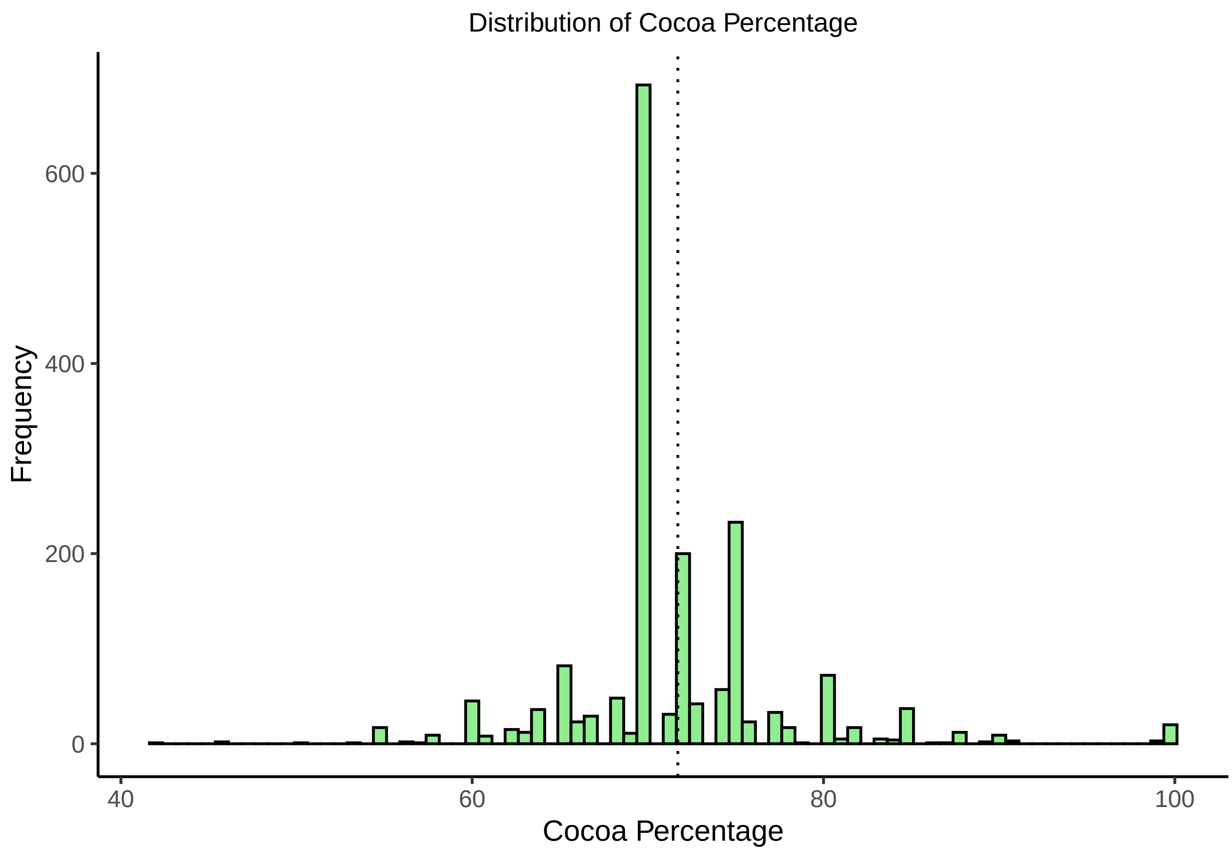
```
# Distribution of Chocolate Bar Rating Class
cleandata %>%
ggplot(aes(RatingClass)) +
geom_bar(color = "black", fill = "pink") +
labs(title = "Distribution of Chocolate Bar Rating Class",
     x = "Scale of Rating",y = "Frequency") +
theme_classic()+
theme(plot.title = element_text(size = 13, color = "black", hjust = 0.5),
axis.text.x = element_text(size = 8, angle = 0, hjust = 0.5))
```



The figure *"Distribution of Chocolate Bar Rating Class"* also depict the rating distribution and we can clearly observe that most of the chocolate bar ratings are *"Satisfactory"*.

Distribution of Chocolate Bar Ratings by percentage of cocoa

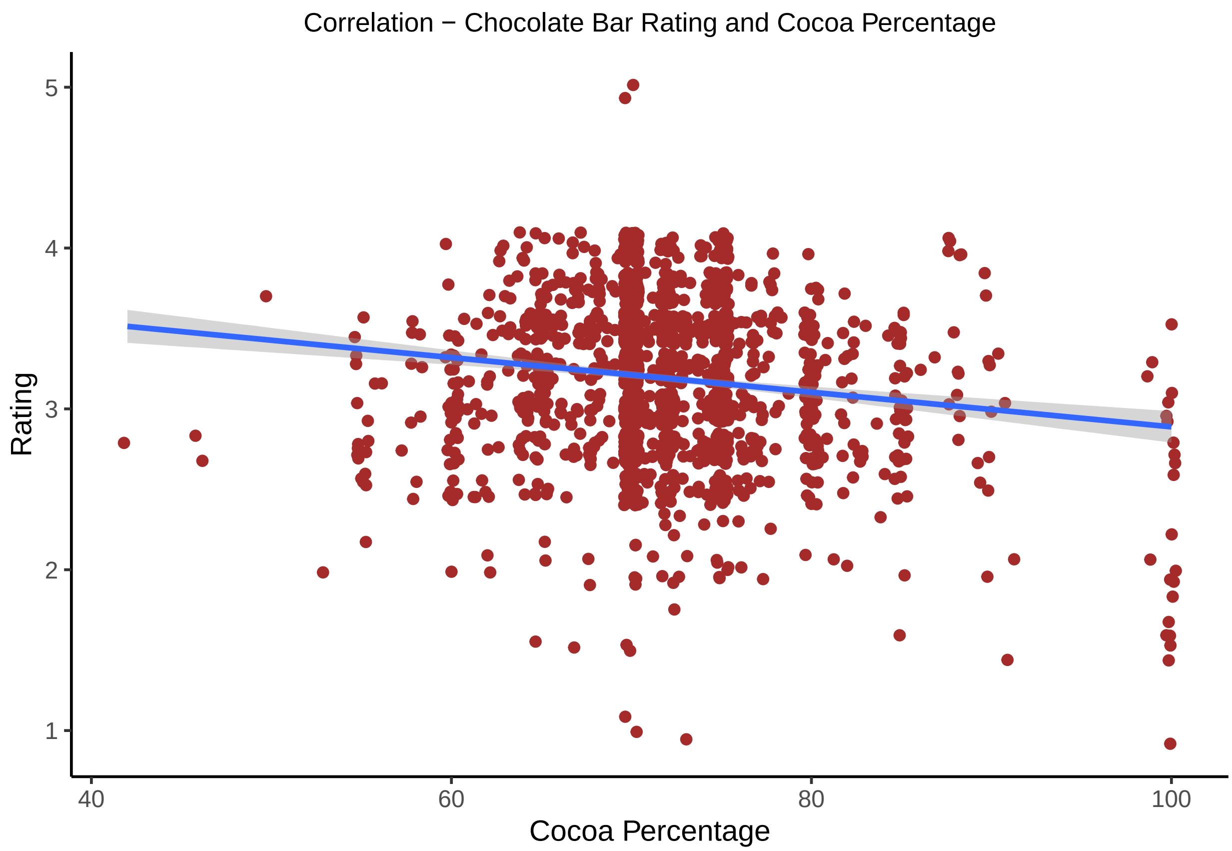
```
# Distribution of Chocolate Bar Ratings by percentage of cocoa
cocoapercent <- mean(cleandata$CocoaPercentage)

cleandata %>%
ggplot(aes(CocoaPercentage)) +
geom_histogram(color = "black", fill = "lightgreen", binwidth = 0.75) +
geom_vline(xintercept = cocoapercent, col = "black", linetype = "dotted") +
labs(title = "Distribution of Cocoa Percentage", x = "Cocoa Percentage", y = "Frequency") +
theme_classic() +
theme(plot.title = element_text(size = 10, color = "black", hjust = 0.5))
```



The figure *"Distribution of Cocoa Percentage"* shows the rating distribution on the basis of percentage of cocoa present. The vertical dashed line represents the average percentage of cocoa μ (***71.706545***) across all Chocolate bars and we can clearly observe from the figure that most of the chocolate bars are made with around 65% and 75% of cacao in the bar, with around 700 chocolate bars being made with 70% cacao.

To check the presence of any correlation between Chocolate Bar Ratings and percentage of cocoa

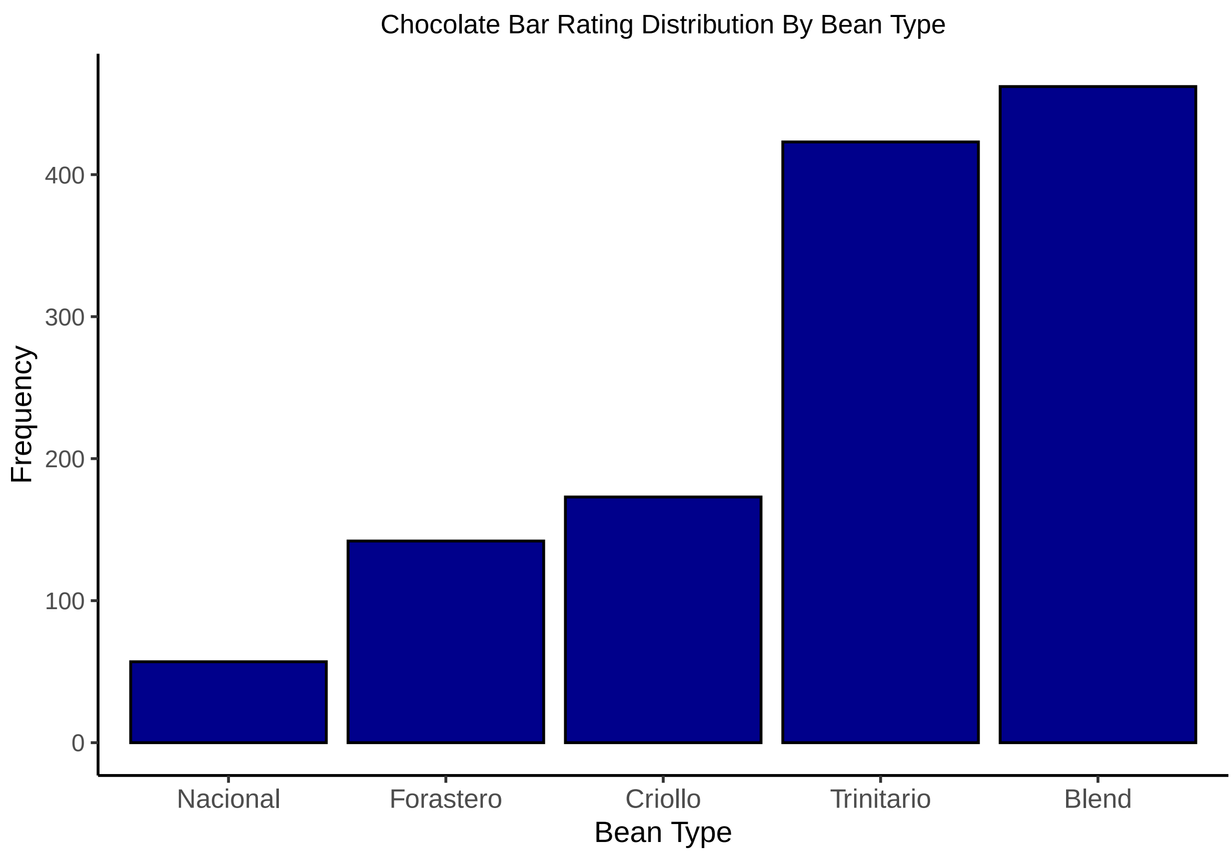
```
# To check the presence of any correlation between Chocolate Bar Ratings and percentage of cocoa
cleandata %>%
ggplot(aes(y = Rating, x = CocoaPercentage)) +
geom_jitter(color = "brown") +
geom_smooth(method = "lm") +
labs(title = "Correlation - Chocolate Bar Rating and Cocoa Percentage",
     x = "Cocoa Percentage",y = "Rating") +
theme_classic() +
theme(plot.title = element_text(size = 10, color = "black", hjust = 0.5))
```



Hence, from figure *"Correlation - Chocolate Bar Rating and Cocoa Percentage"* we observe that there is inverse relation between percentage of cocoa and ratings, i.e. when the percentage of cocoa increases, the rating of the chocolate decreases mildly.

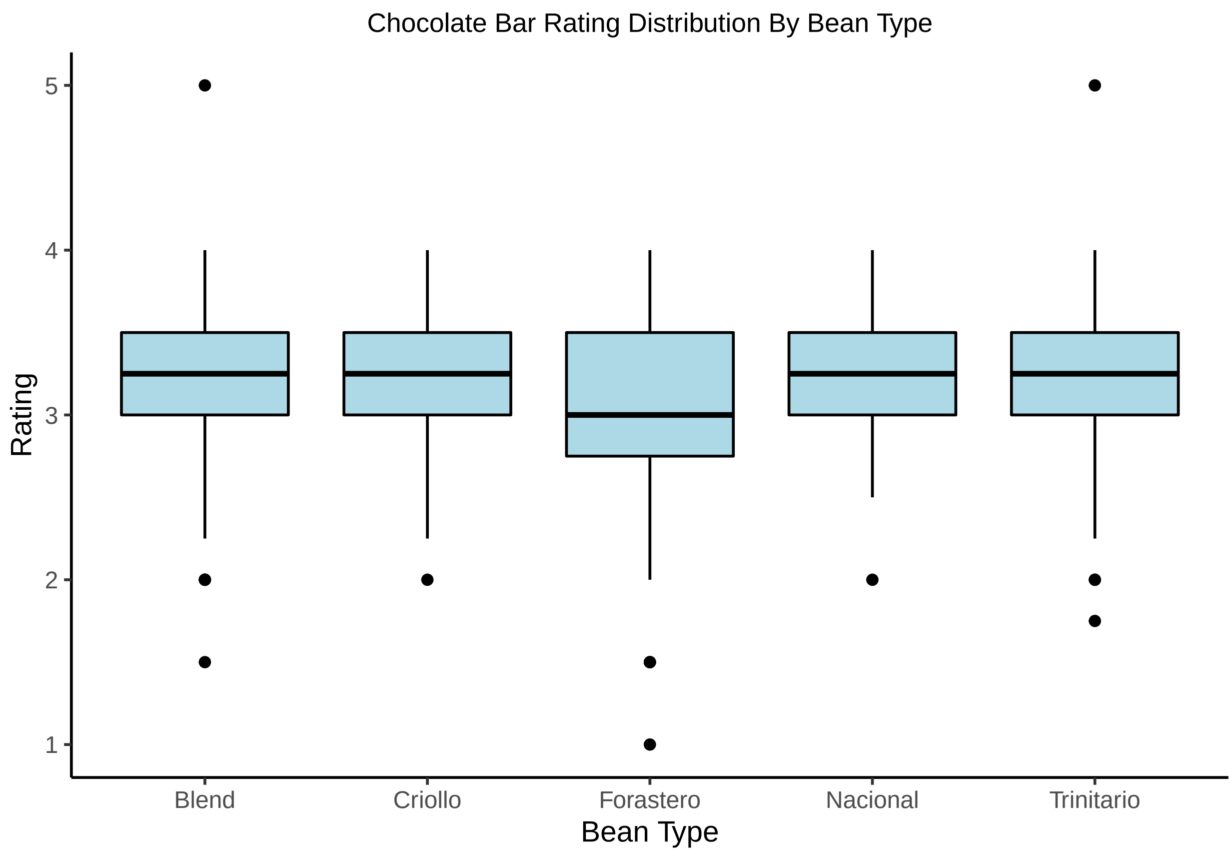
Distribution of Chocolate Bar Ratings on the basis of Bean Type

```
# Distribution of Chocolate Bar Ratings on the basis of Bean Type
cleandata%>%
filter(!is.na(BeanType)) %>%
group_by(BeanType) %>%
summarize(Rating_Count = n(), Rating_Average = mean(Rating)) %>%
ggplot(aes(x = reorder(BeanType, Rating_Count), y = Rating_Count)) +
geom_bar(stat = "identity", color = "black", fill = "darkblue") +
labs(title = "Chocolate Bar Rating Distribution By Bean Type",
     x = "Bean Type", y = "Frequency") +
theme_classic() +
theme(plot.title = element_text(size = 10, color = "black", hjust = 0.5),
      axis.text.x = element_text(size = 10, angle = 0, hjust = 0.5))
```



The figure *"Chocolate Bar Rating Distribution By Bean Type"* depicts the distribution of rating on the basis of type of bean. We can clealry observe that Blend and Trinitario are the most bean types used. However, it is worth noting that some company manufacturing the chocolate bars do not provide the Type of Bean in order to preserve their recipe.

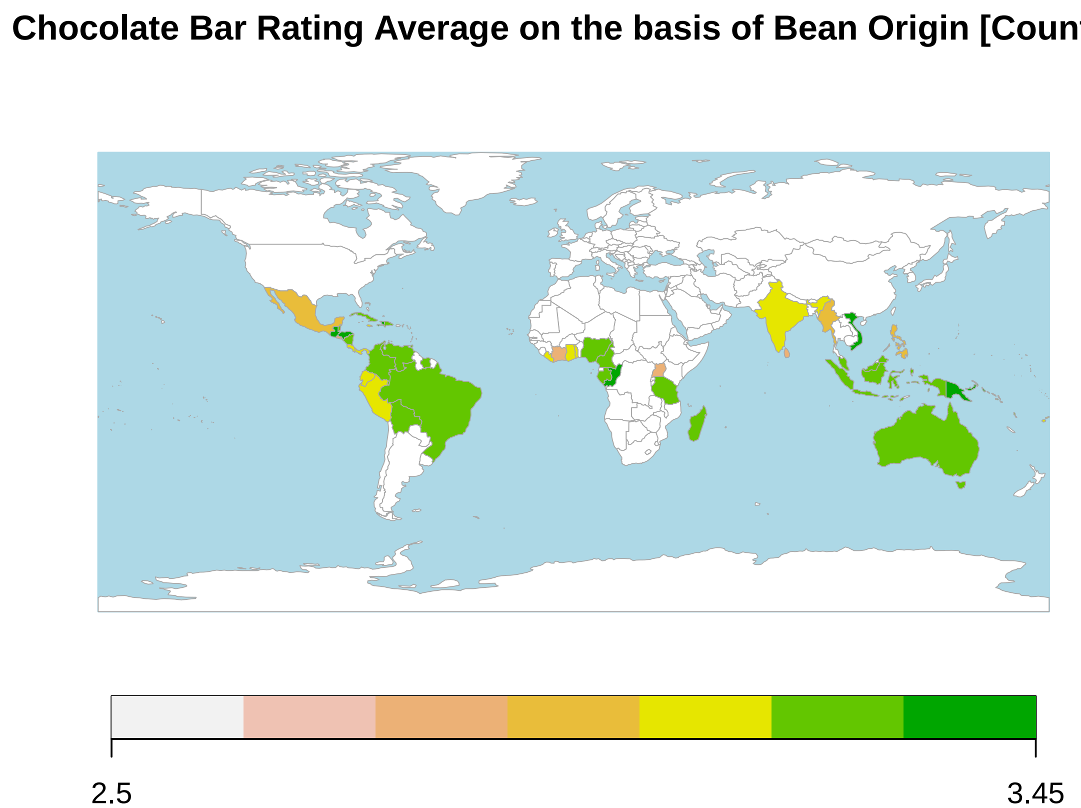
```r
# Distribution of Chocolate Bar Ratings on the basis of Bean Type
cleandata %>%
filter(!is.na(BeanType)) %>%
ggplot(aes(x = BeanType, y = Rating)) +
geom_boxplot(color = "black", fill = "lightblue") +
labs(title = "Chocolate Bar Rating Distribution By Bean Type", x = "Bean Type", y = "Rating") +
theme_classic() +
theme(plot.title = element_text(size = 10, color = "black", hjust = 0.5))
```



There is no difference in the distribution of data on the basis of the type of the bean as shown by the boxplot *"Chocolate Bar Rating Distribution By Bean Type"*, and hence, there is no strong correlation between Ratings and Bean Type.

Chocolate Bar Rating Average on the basis of Bean Origin [Country]

```r
# Chocolate Bar Rating Average on the basis of Bean Origin [Country]
mapCountryData(mapToPlot = BeanOriginMap, nameColumnToPlot="Rating_Average",
               oceanCol = 'lightblue', missingCountryCol = 'white',
               borderCol = 'darkgrey', colourPalette = "terrain",
               mapTitle = "Chocolate Bar Rating Average on the basis of Bean Origin [Country]",
               catMethod = "fixedWidth")
```



The map *"Chocolate Bar Rating Average on the basis of Bean Origin [Country]"* depicts the distribution of ratings all over the world based on Origin of Bean and we observe that the most rated chocolate bars have bean originated from Ecuador, Peru, Venezuela, Dominican Republic, and Madagascar.

Top 15 rankings of the Chocolate Bar Rating Average on the basis of Bean Origin [Country]

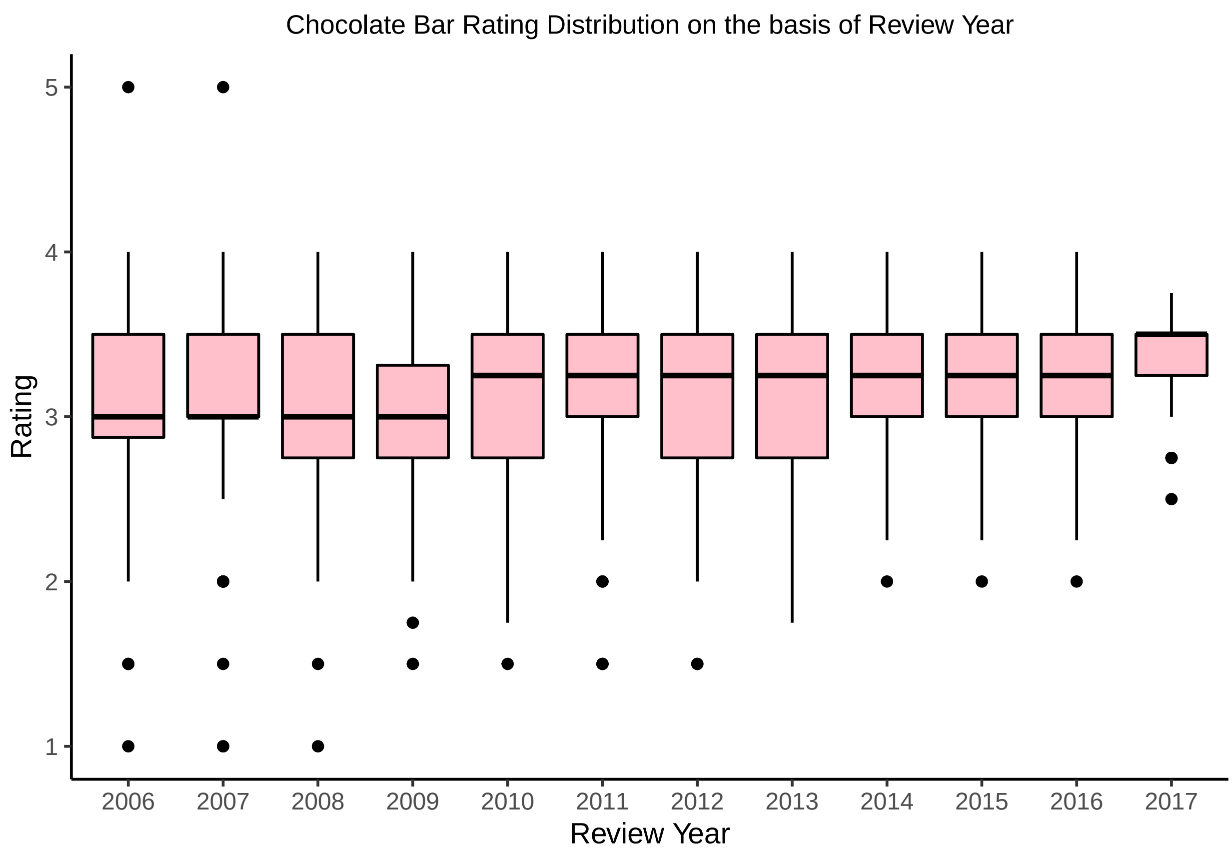
```r
# Top 15 rankings of the Chocolate Bar Rating Average on the basis of Bean Origin [Country]
cleandata%>%
filter(!is.na(BeanOrigin)) %>%
group_by(BeanOrigin) %>%
summarize(Rating_Count = n(), Rating_Average = mean(Rating)) %>%
filter(Rating_Count >= 10) %>%
arrange(desc(Rating_Average)) %>%
head(15) %>%
kable() %>%
kable_styling(bootstrap_options = ("bordered"),font_size = 10,
              position = "center", full_width = FALSE, latex_options = "hold_position")
```

BeanOrigin	Rating_Count	Rating_Average
Haiti	10	3.45000
Honduras	15	3.35000
Guatemala	29	3.34483
Papua New Guinea	47	3.32979
Republic of the Congo	10	3.32500
Vietnam	38	3.31579
Indonesia	21	3.29762
Brazil	59	3.29237
Madagascar	156	3.26923
Cuba	11	3.25000
Venezuela	228	3.24671
Belize	50	3.24500
South Pacific	29	3.24138
Dominican Republic	177	3.22458
Bolivia	58	3.21121

The table above depicts the top 15 rankings of the Chocolate Bar Rating Average on the basis of Bean Origin [Country]

Distribution of Chocolate Bar Ratings on the basis of Review Year

```
# Distribution of Chocolate Bar Ratings on the basis of Review Year
cleandata %>%
filter(!is.na(ReviewYear)) %>%
ggplot(aes(x = ReviewYear,y = Rating)) +
geom_boxplot(color = "black", fill = "pink") +
labs(title = "Chocolate Bar Rating Distribution on the basis of Review Year",
     x = "Review Year", y = "Rating") +
theme_classic() +
theme(plot.title = element_text(size = 10, color = "black", hjust = 0.5))
```



The figure *"Chocolate Bar Rating Distribution By Review Year"* shows that the number of chocolate bars ratings varies from year to year. But, the most of the chocolate ratings are distributed around the average. Also, we notice that there is less variation in the ratings in the recent years.

Top 15 Chocolate Bar Rating Average by Company Name

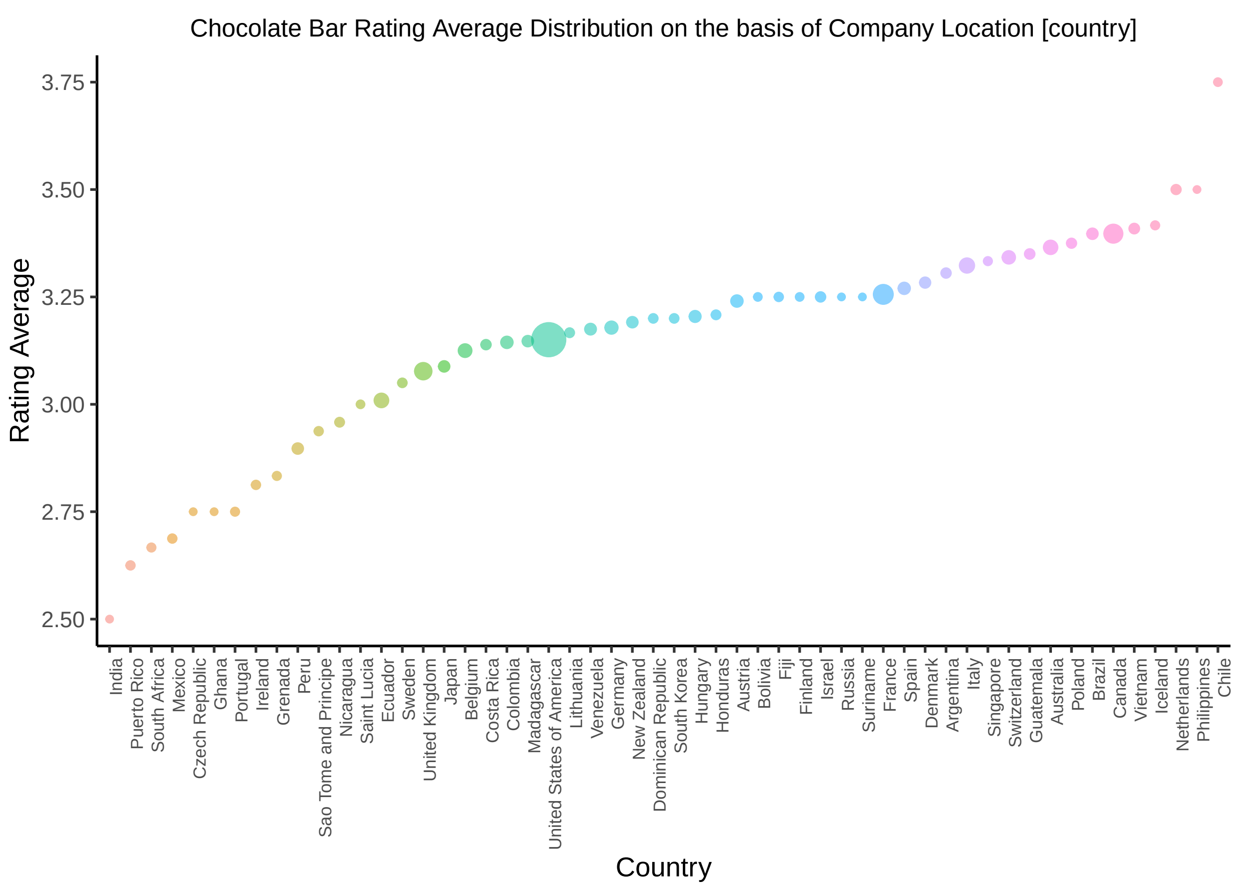
```
# Top 15 Chocolate Bar Rating Average by Company Name
cleandata %>%
group_by(CompanyName, CompanyCountry) %>%
summarize(Rating_Count = n(), Rating_Average = mean(Rating)) %>%
filter(Rating_Count >= 10) %>%
arrange(desc(Rating_Average)) %>%
head(15) %>%
kable() %>%
kable_styling(bootstrap_options = ("bordered"),font_size = 8,
              position = "center", full_width = FALSE, latex_options = "hold_position")
```

CompanyName	CompanyCountry	Rating_Count	Rating_Average
Amedei	Italy	13	3.84615
Idilio (Felchlin)	Switzerland	10	3.77500
Soma	Canada	67	3.66418
Arete	United States of America	22	3.53409
Smooth Chocolator, The	Australia	16	3.51562
Duffy's	United Kingdom	13	3.50000
Pierre Marcolini	Belgium	16	3.50000
Domori	Italy	22	3.47727
Bonnat	France	31	3.47581
Marou	Vietnam	10	3.45000
Sirene	Canada	11	3.40909
Rogue	United States of America	16	3.40625
Szanto Tibor	Hungary	15	3.40000
Fresco	United States of America	26	3.38462
A. Morin	France	23	3.38043

The table depicts the top 15 Chocolate Bar Rating Average by Company Name

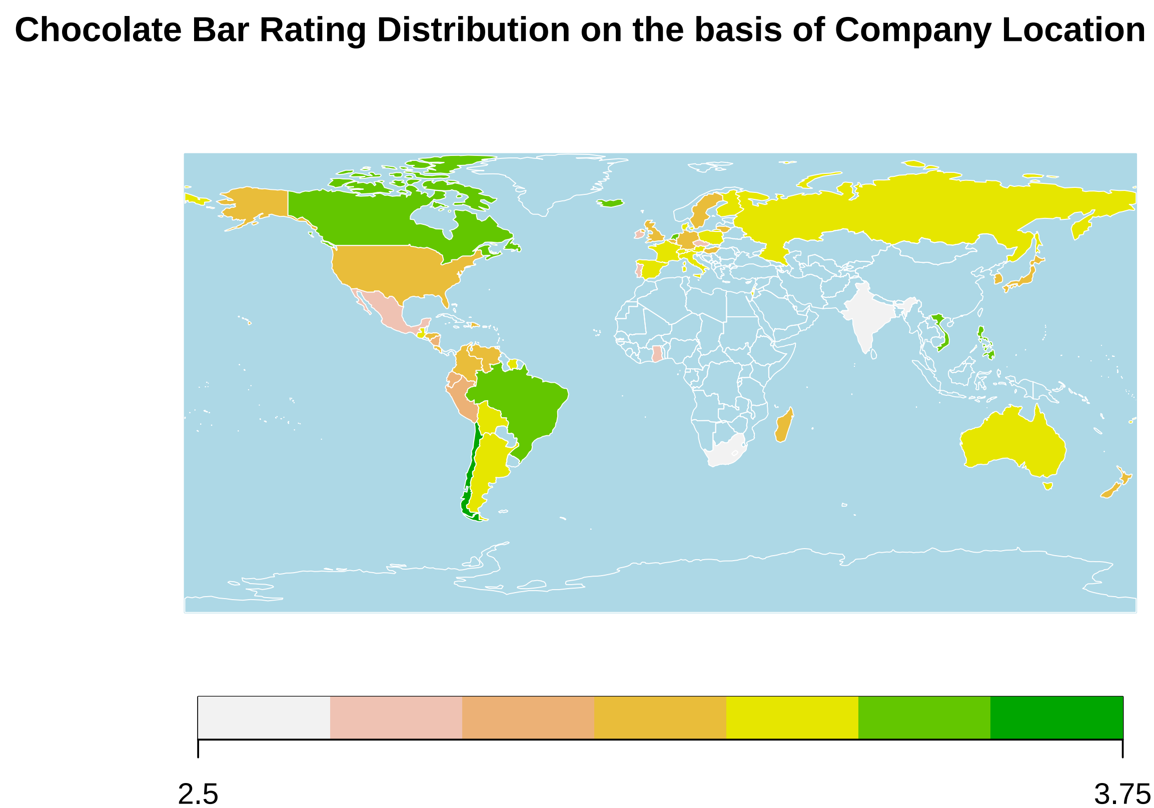
Distribution of Chocolate Bar Rating Average on the basis of location of the company [Country]

```r
# Distribution of Chocolate Bar Rating Average on the basis of location of the company [Country]
cleandata%>%
group_by(CompanyCountry) %>%
summarise(Rating_Count = n(), Rating_Average = mean(Rating)) %>%
arrange(desc(Rating_Average)) %>%
ggplot(aes(y = Rating_Average,x = reorder(CompanyCountry, Rating_Average))) +
geom_point(aes(size = Rating_Count,colour = factor(Rating_Average)),alpha = 0.5) +
labs(title = "Chocolate Bar Rating Average Distribution on the basis of Company Location [country]",
     x = "Country",y = "Rating Average") +
theme_classic() +
theme(plot.title = element_text(size = 10, color = "black", hjust = 0.5),
axis.text.x = element_text(size = 7, angle = 90, hjust = 1),legend.position="none")
```



The figure *"Chocolate Bar Rating Average Distribution on the basis of Company Location [country]"* depicts that ratings of chocolate bars depend on the location of the manufacturing company [country].

```r
# Distribution of Chocolate Bar Ratings Average on the basis of location of the company [Country]
mapCountryData(mapToPlot = CompanyCountryMap, nameColumnToPlot="Rating_Average",
               oceanCol = 'lightblue', borderCol = 'white',
               colourPalette = "terrain",
               mapTitle = "Chocolate Bar Rating Distribution on the basis of Company Location [country]
               catMethod = "fixedWidth")
```



The map *"Chocolate Bar Rating Distribution on the basis of Company Location [country]"* depicts the distribution of ratings of chocolate companies over the world.

Top 15 rankings of the Chocolate Bar Rating Average on the basis of Company Geo-Region

```
# Top 15 rankings of the Chocolate Bar Rating Average on the basis of Company Geo-Region
cleandata %>%
filter(!is.na(CompanyCountry)) %>%
group_by(CompanyCountry) %>%
summarize(Rating_Count = n(), Rating_Average = mean(Rating)) %>%
filter(Rating_Count >= 10) %>%
arrange(desc(Rating_Average)) %>%
head(15) %>%
kable() %>%
kable_styling(bootstrap_options = ("bordered"),font_size = 10,
              position = "center", full_width = FALSE, latex_options = "hold_position")
```

CompanyCountry	Rating_Count	Rating_Average
Vietnam	11	3.40909
Canada	146	3.39726
Brazil	17	3.39706
Australia	52	3.36538
Guatemala	10	3.35000
Switzerland	38	3.34211
Italy	65	3.32308
Denmark	15	3.28333
Spain	25	3.27000
France	168	3.25595
Austria	26	3.24038
Hungary	22	3.20454
New Zealand	17	3.19118
Germany	35	3.17857
Venezuela	20	3.17500

We can observe from the table above that the companies from Vietnam, Brazil and Canada are the most rated companies.

Building Models

Post initial data exploration, we will now build, train, tes tand validate models to get the best accuracy on the validation set. As observed earlier, there are different variables that affect the ratings and to predict the `RatingClass`, we will be using the following:

- `CocoaPercentage`
- `BeanType`
- `BeanOrigin` (Country)
- `CompanyCountry`
- `ReviewYear`

```
############################################################
#  FILTERING AND CREATING THE FINAL DATASET
############################################################

# Create final Dataset
finaldataset <- cleandata %>%
select(CocoaPercentage, BeanType, BeanOrigin, CompanyCountry, ReviewYear,RatingClass) %>%
drop_na()
```

To build and train the different models using different methods, we will conduct the following steps in each method.

- Split our `cleandata` into two datasets: Training set `trainingset` (70%) and Validation set `validationset` (30%).
- Define and Build the model.
- Train the algorithm in the training set `trainingset`.
- Validate the algorithm by running the predictions in the validation dataset `validationset`.
- Iterate over the models until goal is achieved.

```r
###########################################################################
# SPLITTING THE DATASET INTO TRAINING SET (70%) AND VALIDATION SET (30%)
###########################################################################

# First we create Data Partition Index
set.seed(1111)
indexsample <- createDataPartition(y = finaldataset$RatingClass, times = 1, p = 0.7,
                                   list = FALSE)

# Next, we create Training set
trainingset <- finaldataset[indexsample, ]

# Now, we create the Validation set
validationset <- finaldataset[-indexsample, ]

# Finally, we remove unwanted data
rm(datachocolate,indexsample)



#########################################################################
# TRAINING AND VALIDATION
#########################################################################

# 011-01 Configure the number of K-folds for cross validation (Repeated CV)
control <- trainControl(method = "repeatedcv", number = 10, repeats = 3)
```

Our goal is to predict with the highest accuracy - the `RatingClass` - of the Chocolate Bars. We will be using the following methods/models:

1. Support Vector Machine (SVM)
2. Random Forest (RF)
3. Learning Vector Quantization (LVQ)
4. Stochastic Gradient Boosting Machine (GBM)

1. Support Vector Machine (SVM)

An SVM model is a representation of different classes in a hyperplane in multidimensional space. The hyperplane will be generated in an iterative manner by SVM so that the error can be minimized.

The equation of the support vector classifier is as follows:

$$f(x) = \beta_0 + \sum_{i \in S} \alpha_i K(x_i, y_i)$$

where S are the support vectors, α is a weight value which is zero for non support vectors and non-zero for all support vectors, and, $K(x_i, y_i)$ is the Kernel Function that will use the "Radial kernel".

$$K(x, y) = exp(-\gamma \sum_{j=1}^{p} (x_{ij} \breve{} y_{ij})^2)$$

We use this method on our dataset and apply the above steps to achieve and verify the accuracy on the `validationset` set.

```
##########################################################################
# METHOD 1
# SUPPORT VECTOR MACHINE
##########################################################################

method1 <- "SVM"
method1d <- "Support Vector Machine"

# Train on training set
set.seed(1111)
method1train <- train(RatingClass ~ ., data = trainingset, trControl = control,
                      method = "svmRadial")

# Results on the training set
method1results <- method1train$results

method1results %>%
kable() %>%
kable_styling(bootstrap_options = ("bordered"),font_size = 9, position = "center",
              full_width = FALSE, latex_options = "hold_position")
```

sigma	C	Accuracy	Kappa	AccuracySD	KappaSD
0.073593	0.25	0.712994	0.000000	0.006346	0.000000
0.073593	0.50	0.712994	0.000000	0.006346	0.000000
0.073593	1.00	0.716465	0.028691	0.010111	0.039654

```
# Best Accuracy Measure on the training set
accuracy_method1train <- max(method1train$results["Accuracy"])

# Prediction on training set
predict_method1 <- predict(method1train, newdata = validationset)

# Confusion Matrix
confusionmatrix_method1 <- confusionMatrix(predict_method1, validationset$RatingClass)
```

```r
# Results of final model on Validation set
predictresults_method1 <- confusionmatrix_method1$overall

predictresults_method1 %>%
kable(col.names = c("Measure Value")) %>%
kable_styling(bootstrap_options = ("bordered"),font_size = 9, position = "center",
              full_width = FALSE, latex_options = "hold_position")
```

	Measure Value
Accuracy	0.717808
Kappa	0.023811
AccuracyLower	0.668624
AccuracyUpper	0.763419
AccuracyNull	0.715068
AccuracyPValue	0.480195
McnemarPValue	NaN

```r
# Best Accuracy Measure from the Model on Validation set
predictaccuracy_method1 <- predictresults_method1["Accuracy"]

# We create a table to record our approaches and the measure
finalresult_method1 <- tibble(ModelID = method1,
                         ModelMethod = method1d,
                         AccuracyOnTraining = accuracy_method1train,
                         AccuracyOnValidation = predictaccuracy_method1)

# Next, we create a table to record the results
summaryresult <- finalresult_method1

# Finally, we display the summary
summaryresult %>%
kable() %>%
kable_styling(bootstrap_options = ("bordered"),font_size = 9, position = "center",
              full_width = FALSE, latex_options = "hold_position") %>%
column_spec(1, width = "5em") %>%
column_spec(2, width = "20em") %>%
column_spec(4, bold = TRUE)
```

ModelID	ModelMethod	AccuracyOnTraining	AccuracyOnValidation
SVM	Support Vector Machine	0.716465	**0.717808**

The predicted ***Accuracy*** on the `validationset` dataset for the ***Support Vector Machine*** is ***71.780822***.

2. Random Forest (RF)

Random Forest builds multiple decision trees and merges them together to get a more accurate and stable prediction. It builds a 'forest' which is an ensemble of Decision trees.

We use this method on our dataset and apply the above steps to achieve and verify the accuracy on the `validationset` set.

```
##########################################################################
# METHOD 2
# RANDOM FOREST
##########################################################################

method2 <- "RF"
method2d <- "Random Forest"

# Train on training set
set.seed(1111)
method2train <- train(RatingClass ~ ., data = trainingset, trControl = control,
                      method = "rf")

# Results on the training set
method2results <- method2train$results

method2results %>%
kable() %>%
kable_styling(bootstrap_options = ("bordered"), font_size = 9, position = "center",
              full_width = FALSE, latex_options = "hold_position")
```

mtry	Accuracy	Kappa	AccuracySD	KappaSD
2	0.713706	0.000000	0.006160	0.000000
61	0.683321	0.126750	0.038479	0.098853
121	0.668252	0.111074	0.041331	0.092794

```
# Best Accuracy Measure on the training set
accuracy_method2train <- max(method2train$results["Accuracy"])


# Prediction on training set
predict_method2 <- predict(method2train, newdata = validationset)

# Confusion Matrix
confusionmatrix_method2 <- confusionMatrix(predict_method2, validationset$RatingClass)

# Results of final model on Validation set
predictresults_method2 <- confusionmatrix_method2$overall

predictresults_method2 %>%
kable(col.names = c("Measure Value")) %>%
kable_styling(bootstrap_options = ("bordered"), font_size = 9, position = "center",
              full_width = FALSE, latex_options = "hold_position")
```

```
# Best Accuracy Measure from the Model on Validation set
predictaccuracy_method2 <- predictresults_method2["Accuracy"]

# We create a table to record our approaches and the measure
```

	Measure Value
Accuracy	0.715068
Kappa	0.000000
AccuracyLower	0.665773
AccuracyUpper	0.760836
AccuracyNull	0.715068
AccuracyPValue	0.526415
McnemarPValue	NaN

```
finalresult_method2 <- tibble(ModelID = method2,
                              ModelMethod = method2d,
                              AccuracyOnTraining = accuracy_method2train,
                              AccuracyOnValidation = predictaccuracy_method2)

# Next, we create a table to record the results
summaryresult <- bind_rows(summaryresult, finalresult_method2)

# Finally, we display the summary
summaryresult %>%
kable() %>%
kable_styling(bootstrap_options = ("bordered"), font_size = 9, position = "center",
              full_width = FALSE, latex_options = "hold_position") %>%
column_spec(1, width = "5em") %>%
column_spec(2, width = "20em") %>%
column_spec(4, bold = TRUE)
```

ModelID	ModelMethod	AccuracyOnTraining	AccuracyOnValidation
SVM	Support Vector Machine	0.716465	**0.717808**
RF	Random Forest	0.713706	**0.715068**

The predicted ***Accuracy*** on the `validationset` dataset for the ***Random Forest*** is about ***71.506849***.

3. Learning Vector Quantization (LVQ)

The Learning Vector Quantization algorithm (LVQ) is an is a prototype-based supervised classification algorithm that lets you choose how many training instances to hang onto and learns exactly what those instances should look like.

We use this method on our dataset and apply the above steps to achieve and verify the accuracy on the `validationset` set.

```
##########################################################################
# METHOD 3
# LEARNING VECTOR QUANTIZATION (LVQ)
##########################################################################

method3 <- "LVQ"
method3d <- "Learning Vector Quantization"

# Train on training set
set.seed(1111)
method3train <- train(RatingClass ~ ., data = trainingset, trControl = control,
                      method = "lvq")

# Results on the training set
method3results <- method3train$results

method3results %>%
kable() %>%
kable_styling(bootstrap_options = ("bordered"), font_size = 9, position = "center",
              full_width = FALSE, latex_options = "hold_position")
```

size	k	Accuracy	Kappa	AccuracySD	KappaSD
129	1	0.676742	0.079843	0.031430	0.087557
129	6	0.695032	0.088405	0.031195	0.091002
129	11	0.700153	0.017116	0.017084	0.052977
193	1	0.665041	0.090829	0.034780	0.086484
193	6	0.690769	0.068104	0.028698	0.073083
193	11	0.704048	0.032464	0.018698	0.054047
258	1	0.678694	0.103742	0.031554	0.086438
258	6	0.693095	0.064163	0.024236	0.078498
258	11	0.701735	0.021861	0.016449	0.051722

```
# Best Accuracy Measure on the training set
accuracy_method3train <- max(method3train$results["Accuracy"])


# Prediction on training set
predict_method3 <- predict(method3train, newdata = validationset)

# Confusion Matrix
confusionmatrix_method3 <- confusionMatrix(predict_method3, validationset$RatingClass)

# Results of final model on Validation set
predictresults_method3 <- confusionmatrix_method3$overall

predictresults_method3 %>%
kable(col.names = c("Measure Value")) %>%
```

```r
kable_styling(bootstrap_options = ("bordered"), font_size = 9, position = "center",
              full_width = FALSE, latex_options = "hold_position")
```

	Measure Value
Accuracy	0.676712
Kappa	-0.005228
AccuracyLower	0.626068
AccuracyUpper	0.724461
AccuracyNull	0.715068
AccuracyPValue	0.952298
McnemarPValue	NaN

```r
# Best Accuracy Measure from the Model on Validation set
predictaccuracy_method3 <- predictresults_method3["Accuracy"]

# We create a table to record our approaches and the measure
finalresult_method3 <- tibble(ModelID = method3,
                              ModelMethod = method3d,
                              AccuracyOnTraining = accuracy_method3train,
                              AccuracyOnValidation = predictaccuracy_method3)

# Next, we create a table to record the results
summaryresult <- bind_rows(summaryresult, finalresult_method3)

# Finally, we display the summary
summaryresult %>%
kable() %>%
kable_styling(bootstrap_options = ("bordered"), font_size = 9, position = "center",
              full_width = FALSE, latex_options = "hold_position")%>%
column_spec(1, width = "5em") %>%
column_spec(2, width = "20em") %>%
column_spec(4, bold = TRUE)
```

ModelID	ModelMethod	AccuracyOnTraining	AccuracyOnValidation
SVM	Support Vector Machine	0.716465	**0.717808**
RF	Random Forest	0.713706	**0.715068**
LVQ	Learning Vector Quantization	0.704048	**0.676712**

The predicted ***Accuracy*** on the `validationset` dataset for the ***Learning Vector Quantization*** is about ***67.671233***.

4. Stochastic Gradient Boosting Machine (GBM)

Gradient boosting is a machine learning technique in which the ensembles are constructed from decision tree models. Trees are added one at a time to the ensemble and fit to correct the prediction errors made by prior models.

We use this method on our dataset and apply the above steps to achieve and verify the accuracy on the `validationset` set.

```
###########################################################################
# METHOD 4
# STOCHASTIC GRADIENT BOOSTING MACHINE (GBM)
###########################################################################

method4 <- "GBM"
method4d <- "Stochastic Gradient Boosting Machine"

# Train on training set
set.seed(1111)
method4train <- train(RatingClass ~ ., data = trainingset, trControl = control,
                      method = "gbm", verbose = FALSE)

# Results on the training set
method4results <- method4train$results

method4results %>%
kable() %>%
kable_styling(bootstrap_options = ("bordered"), font_size = 9, position = "center",
              full_width = FALSE, latex_options = "hold_position")
```

	shrinkage	interaction.depth	n.minobsinnode	n.trees	Accuracy	Kappa	AccuracySD	KappaSD
1	0.1	1	10	50	0.701307	0.004655	0.019449	0.044967
4	0.1	2	10	50	0.696236	0.015897	0.022352	0.049131
7	0.1	3	10	50	0.700491	0.042745	0.022070	0.061316
2	0.1	1	10	100	0.694275	0.006782	0.021006	0.051437
5	0.1	2	10	100	0.688046	0.022745	0.027086	0.067070
8	0.1	3	10	100	0.683372	0.040700	0.030114	0.077101
3	0.1	1	10	150	0.696218	0.034159	0.021292	0.064712
6	0.1	2	10	150	0.686090	0.024271	0.026467	0.066660
9	0.1	3	10	150	0.686491	0.063941	0.028295	0.070802

```
# Best Accuracy Measure on the training set
accuracy_method4train <- max(method4train$results["Accuracy"])

# Prediction on training set
predict_method4 <- predict(method4train, newdata = validationset)

# Confusion Matrix
confusionmatrix_method4 <- confusionMatrix(predict_method4, validationset$RatingClass)

# Results of final model on Validation set
predictresults_method4 <- confusionmatrix_method4$overall

predictresults_method4 %>%
kable(col.names = c("Measure Value")) %>%
kable_styling(bootstrap_options = ("bordered"), font_size = 9, position = "center",
```

```
              full_width = FALSE, latex_options = "hold_position")
```

	Measure Value
Accuracy	0.720548
Kappa	0.064573
AccuracyLower	0.671477
AccuracyUpper	0.766000
AccuracyNull	0.715068
AccuracyPValue	0.434151
McnemarPValue	NaN

```
# Best Accuracy Measure from the Model on Validation set
predictaccuracy_method4 <- predictresults_method4["Accuracy"]

# We create a table to record our approaches and the measure
finalresult_method4 <- tibble(ModelID = method4,
                         ModelMethod = method4d,
                         AccuracyOnTraining = accuracy_method4train,
                         AccuracyOnValidation = predictaccuracy_method4)

# Next, we create a table to record the results
summaryresult <- bind_rows(summaryresult,finalresult_method4)

# Finally, we display the summary
summaryresult %>%
kable() %>%
kable_styling(bootstrap_options = ("bordered"), font_size = 9, position = "center",
              full_width = FALSE, latex_options = "hold_position") %>%
column_spec(1, width = "5em") %>%
column_spec(2, width = "20em") %>%
column_spec(4, bold = TRUE)
```

ModelID	ModelMethod	AccuracyOnTraining	AccuracyOnValidation
SVM	Support Vector Machine	0.716465	**0.717808**
RF	Random Forest	0.713706	**0.715068**
LVQ	Learning Vector Quantization	0.704048	**0.676712**
GBM	Stochastic Gradient Boosting Machine	0.701307	**0.720548**

The predicted ***Accuracy*** on the `validationset` dataset for the ***Stochastic Gradient Boosting Machine*** is about ***72.054795***.

3. Results

Here is the summary of the Accuracy measures after building, training and validating different models on the `validationset` dataset:

```
################################################
# RESULTS SUMMARY
################################################

# Result Summary
summaryresult %>%
arrange(desc(AccuracyOnValidation), desc(AccuracyOnTraining)) %>%
kable() %>%
kable_styling(bootstrap_options = ("bordered"), font_size = 10, position = "center",
              full_width = FALSE, latex_options = "hold_position") %>%
column_spec(1, width = "5em") %>%
column_spec(2, width = "20em") %>%
column_spec(4, bold = TRUE)
```

ModelID	ModelMethod	AccuracyOnTraining	AccuracyOnValidation
GBM	Stochastic Gradient Boosting Machine	0.701307	**0.720548**
SVM	Support Vector Machine	0.716465	**0.717808**
RF	Random Forest	0.713706	**0.715068**
LVQ	Learning Vector Quantization	0.704048	**0.676712**

Based on the `Accuracy` measure only, we can observe that the model that predict our `Chocolate Bar Rating Class` with the best Accuracy of ***72.054795*** is ***Stochastic Gradient Boosting Machine***.

4. Conclusion

Using various Machine Learning models, we have used the Chocolate Bar Rating System to predict the Chocolate Bar Rating Class in this project.

Beginning with exploring the data, we observe the structure of the dataset and how we can customize and standardize the initial data. At that point, we realise how important it is to pre-process the data to get a better and in depth understanding of the dataset and to have a smooth conduct of the project in order to reach to a conclusion.

Next, we visualised the numerous variables present in the data that directly affect the Chocolate Bar ratings and we observed the importance visualization holds in order to get and better comprehension of the data which can help us to eventually apply appropriate models and methods to fulfil the goal of this project.

Finally, we move further to build, train, test and validate the dataset using numerous methods like Support Vector Machine, Random Forest Model, Learning Vector Quantization and Stochastic Gradient Boosting Machine which were also used to get an accuracy of our predictions.

Since the dataset is not large enough, the size of the dataset is one of the drawbacks to the project. In the end, we conclude that people prefer sweet Chocolate Bars than bitter ones among all of them made from the five bean types listed above in the project. Further, we also observe the correlation that as the percentage of cocoa increases, the ratings of the chocolate bars decrease. Also, concluding the accuracy given by the four models, we observe that ***Stochastic Gradient Boosting Machine*** gives the best accuracy of ***72.054795***.