beer_data_analysis

Pei Yu Lin

Section 1 - data management and statistics

```r
# Read in the data
beer_data <- read_csv("Craft-Beer_data_set.txt")
```

```
## Rows: 5558 Columns: 18
## -- Column specification --------------------------------------------------------------------------------
## Delimiter: ","
## chr  (3): Name, Style, Brewery
## dbl (15): ABV, rating, minIBU, maxIBU, Astringency, Body, Alcohol, Bitter, Sweet, Sour, Salty, F...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```r
# Check data structure
str(beer_data)
```

```
## spec_tbl_df [5,558 x 18] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##  $ Name       : chr [1:5558] "Amber" "Double Bag" "Long Trail Ale" "Doppelsticke" ...
##  $ Style      : chr [1:5558] "Altbier" "Altbier" "Altbier" "Altbier" ...
##  $ Brewery    : chr [1:5558] "Alaskan Brewing Co." "Long Trail Brewing Co." "Long Trail Brewing Co."
##  $ ABV        : num [1:5558] 5.3 7.2 5 8.5 5.3 7.2 6 5.3 5 4.8 ...
##  $ rating     : num [1:5558] 3.65 3.9 3.58 4.15 3.67 3.78 4.1 3.46 3.6 4.1 ...
##  $ minIBU     : num [1:5558] 25 25 25 25 25 25 25 25 25 25 ...
##  $ maxIBU     : num [1:5558] 50 50 50 50 50 50 50 50 50 50 ...
##  $ Astringency: num [1:5558] 13 12 14 13 21 25 22 28 18 25 ...
##  $ Body       : num [1:5558] 32 57 37 55 69 51 45 40 49 35 ...
##  $ Alcohol    : num [1:5558] 9 18 6 31 10 26 13 3 5 4 ...
##  $ Bitter     : num [1:5558] 47 33 42 47 63 44 46 40 37 38 ...
##  $ Sweet      : num [1:5558] 74 55 43 101 120 45 62 58 73 39 ...
##  $ Sour       : num [1:5558] 33 16 11 18 14 9 25 29 22 13 ...
##  $ Salty      : num [1:5558] 0 0 0 1 0 1 1 0 0 1 ...
##  $ Fruits     : num [1:5558] 33 24 10 49 19 11 34 36 21 8 ...
##  $ Hoppy      : num [1:5558] 57 35 54 40 36 51 60 54 37 60 ...
##  $ Spices     : num [1:5558] 8 12 4 16 15 20 4 8 4 16 ...
##  $ Malty      : num [1:5558] 111 84 62 119 218 95 103 97 98 97 ...
##  - attr(*, "spec")=
##   .. cols(
##   ..   Name = col_character(),
```

```
##   ..   Style = col_character(),
##   ..   Brewery = col_character(),
##   ..   ABV = col_double(),
##   ..   rating = col_double(),
##   ..   minIBU = col_double(),
##   ..   maxIBU = col_double(),
##   ..   Astringency = col_double(),
##   ..   Body = col_double(),
##   ..   Alcohol = col_double(),
##   ..   Bitter = col_double(),
##   ..   Sweet = col_double(),
##   ..   Sour = col_double(),
##   ..   Salty = col_double(),
##   ..   Fruits = col_double(),
##   ..   Hoppy = col_double(),
##   ..   Spices = col_double(),
##   ..   Malty = col_double()
##   .. )
##  - attr(*, "problems")=<externalptr>
```

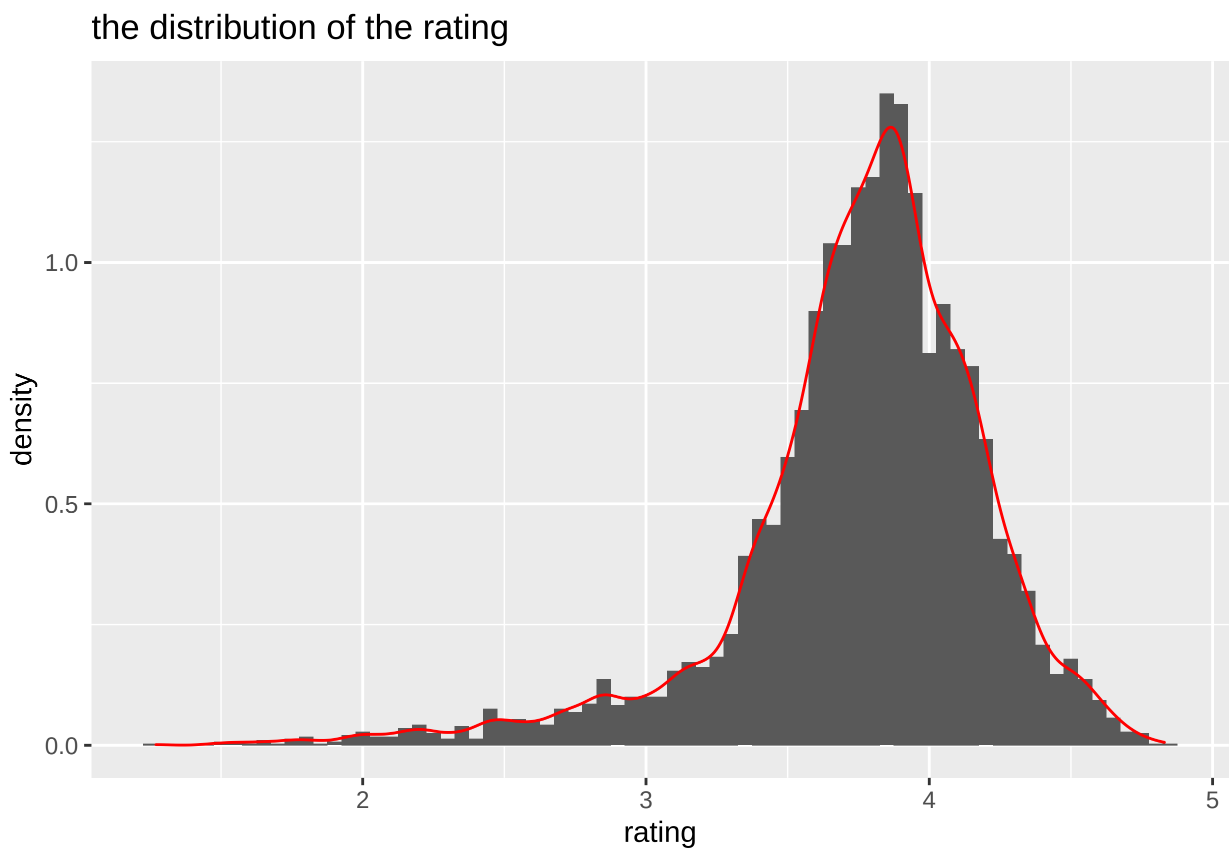
```
# Categorize data into general categories by detecting strings or category
# identifiers in Style column
new_beer_data <- beer_data%>%
  mutate(category = ifelse(Style %like% "IPA", "IPA",
                  ifelse(Style %like% "Lager", "Lager",
                  ifelse(Style %like% "Porter", "Porter",
                  ifelse(Style %like% "Stout", "Stout",
                  ifelse(Style %like% "Wheat", "Wheat",
                  ifelse(Style %like% "Pale", "Pale",
                  ifelse(Style %like% "Pilsner", "Pilsner",
                  ifelse(Style %like% "Bock", "Bock", "Others")))))))))

# Check data value
summary(new_beer_data)
```

```
##       Name              Style              Brewery                ABV              rating
##  Length:5558        Length:5558        Length:5558        Min.   : 0.000   Min.   :1.27
##  Class :character   Class :character   Class :character   1st Qu.: 5.000   1st Qu.:3.59
##  Mode  :character   Mode  :character   Mode  :character   Median : 6.000   Median :3.82
##                                                           Mean   : 6.634   Mean   :3.76
##                                                           3rd Qu.: 7.900   3rd Qu.:4.04
##                                                           Max.   :57.500   Max.   :4.83
##      minIBU          maxIBU         Astringency         Body            Alcohol
##  Min.   : 0.00   Min.   :  0.00   Min.   : 0.00   Min.   :  0.00   Min.   :  0.00
##  1st Qu.:10.00   1st Qu.: 25.00   1st Qu.: 8.00   1st Qu.: 25.00   1st Qu.:  5.00
##  Median :20.00   Median : 35.00   Median :14.00   Median : 38.00   Median : 10.00
##  Mean   :20.72   Mean   : 38.45   Mean   :15.94   Mean   : 42.75   Mean   : 15.98
##  3rd Qu.:25.00   3rd Qu.: 45.00   3rd Qu.:22.00   3rd Qu.: 55.00   3rd Qu.: 20.00
##  Max.   :65.00   Max.   :100.00   Max.   :83.00   Max.   :197.00   Max.   :139.00
##      Bitter           Sweet             Sour             Salty            Fruits
##  Min.   :  0.00   Min.   :  0.00   Min.   :  0.00   Min.   : 0.000   Min.   :  0.00
##  1st Qu.: 13.00   1st Qu.: 27.00   1st Qu.:  9.00   1st Qu.: 0.000   1st Qu.: 10.00
##  Median : 29.00   Median : 49.50   Median : 21.00   Median : 0.000   Median : 28.00
##  Mean   : 34.32   Mean   : 53.63   Mean   : 34.61   Mean   : 1.314   Mean   : 39.38
```

```
##  3rd Qu.: 51.00   3rd Qu.: 74.00   3rd Qu.: 44.00   3rd Qu.: 1.000   3rd Qu.: 61.75
##  Max.   :150.00   Max.   :263.00   Max.   :323.00   Max.   :66.000   Max.   :222.00
##      Hoppy            Spices            Malty           category
##  Min.   :  0.00   Min.   :  0.00   Min.   :  0.00   Length:5558
##  1st Qu.: 14.00   1st Qu.:  4.00   1st Qu.: 33.00   Class :character
##  Median : 30.00   Median :  9.00   Median : 65.00   Mode  :character
##  Mean   : 38.41   Mean   : 17.58   Mean   : 68.59
##  3rd Qu.: 56.00   3rd Qu.: 22.00   3rd Qu.: 99.00
##  Max.   :193.00   Max.   :184.00   Max.   :304.00
```

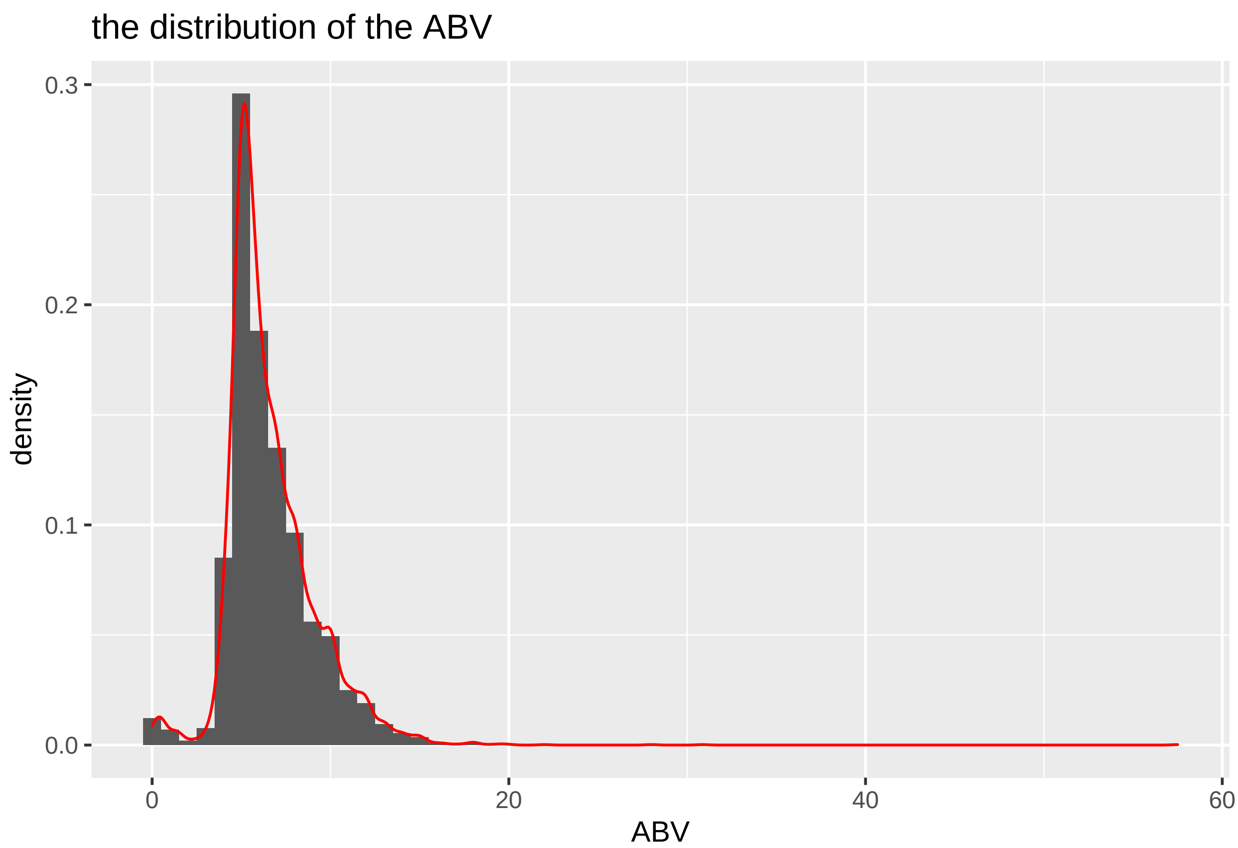
```
# discover outliers in the data set
```

```
# Visualize data and check their distributions

## Rating
rating.plot <- ggplot(new_beer_data, aes(x=rating, y=..density..)) +
  geom_histogram(binwidth = 0.05) + geom_density(col = "red") +
  labs(title="the distribution of the rating")

rating.plot # normal distribution
```

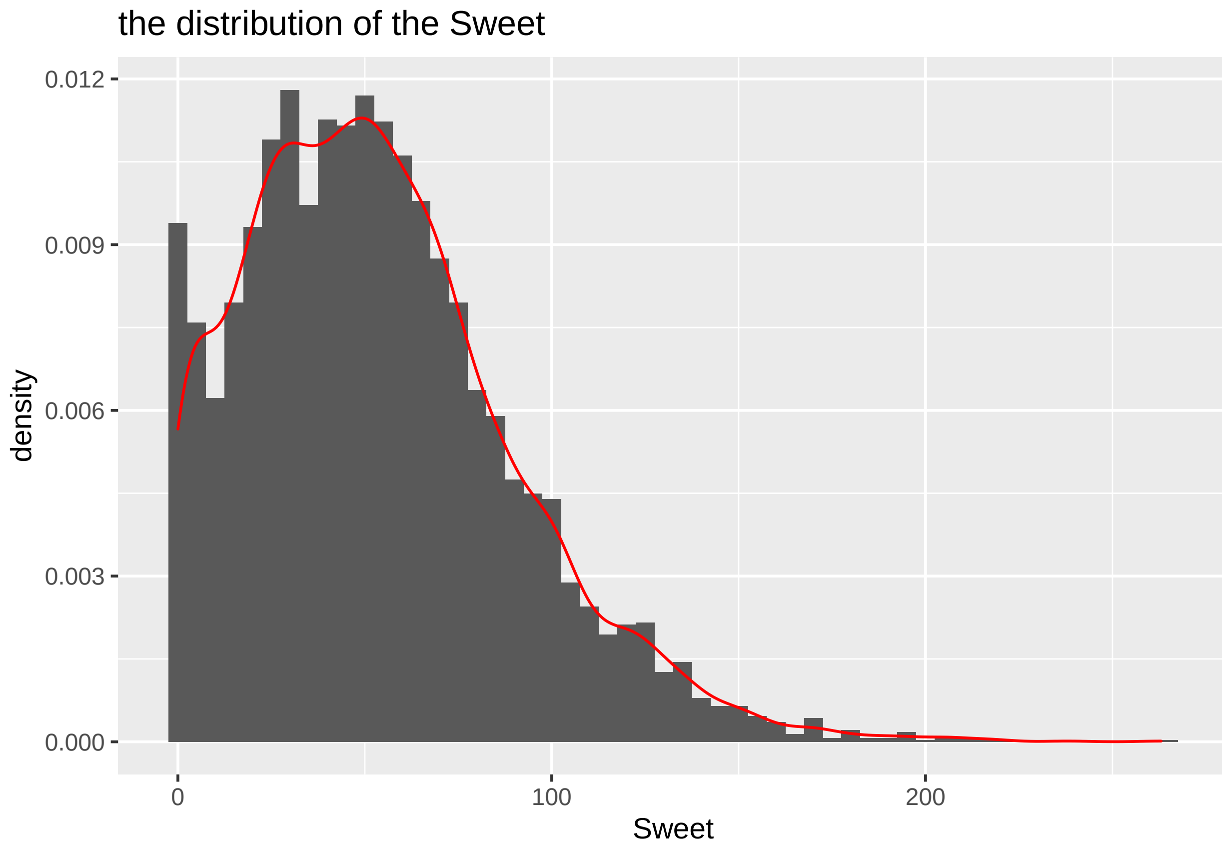



```
# ABV
ABV.plot <- ggplot(new_beer_data, aes(x=ABV, y=..density..)) +
  geom_histogram(binwidth = 1) + geom_density(col="red") +
  labs(title="the distribution of the ABV")
```

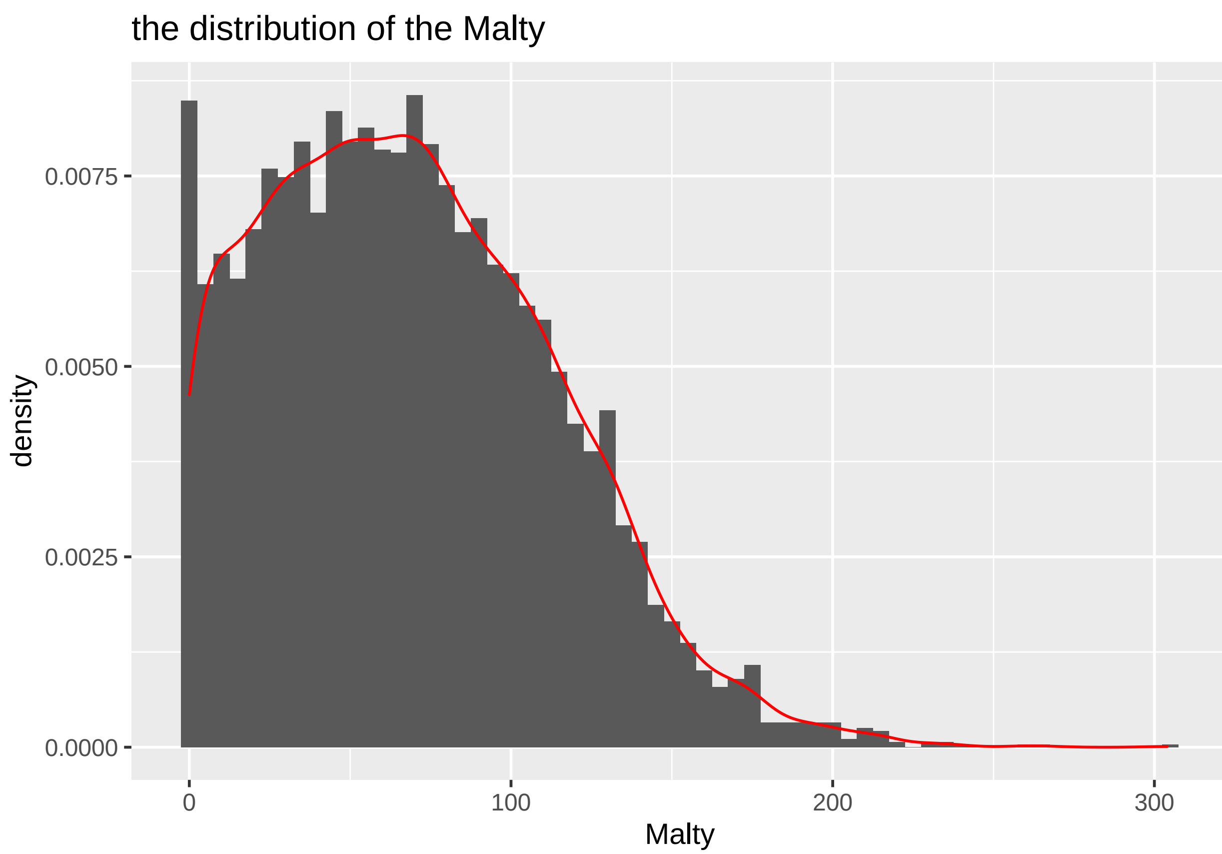
```
ABV.plot # normal distribution
```




```
# Sweet
Sweet.plot <- ggplot(new_beer_data, aes(x=Sweet, y=..density..)) +
  geom_histogram(binwidth = 5) + geom_density(col="red") +
  labs(title="the distribution of the Sweet")

Sweet.plot # normal distribution
```




```
# Malty
Malty.plot <- ggplot(new_beer_data, aes(x=Malty, y=..density..)) +
  geom_histogram(binwidth = 5) + geom_density(col = "red") +
  labs(title="the distribution of the Malty")

Malty.plot #normal distribution and having outliers
```




```r
# ABV
## Assume the value of ABV above 20 is an extreme values and remove the outliers
new_beer_data <- filter(new_beer_data, ABV <= 20)

# Sweet
##Assume the value of Sweet above 200 is an extreme values and
## remove the outliers
new_beer_data <- filter(new_beer_data, Sweet <= 200)

# Malty
## Assume the value of Malty above 250 is an extreme values and
##remove the outliers
new_beer_data <- filter(new_beer_data, Malty <= 250)

# Check the type of variable 'category'
typeof(new_beer_data$category)
```

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

```r
# Change it as a factor
new_beer_data$category <- as.factor(new_beer_data$category)
# Get the final number of each category
summary(new_beer_data$category)
```

```
##     Bock      IPA    Lager   Others    Pale Pilsner  Porter    Stout    Wheat
```

```
##      246      350      900     2599      250      150      297      398      350
```

Calculate the mean rating and 95% confidence interval of the rating within each category using a linear model

```r
lm.rating.category <- lm(rating~category, data=new_beer_data)
emmeans.rating.category <- emmeans(lm.rating.category, ~category)
emmeans.rating.category
```

```
##  category emmean       SE   df lower.CL upper.CL
##  Bock      3.818 0.025322 5531    3.768    3.867
##  IPA       4.029 0.021229 5531    3.988    4.071
##  Lager     3.357 0.013239 5531    3.331    3.383
##  Others    3.806 0.007791 5531    3.791    3.822
##  Pale      3.774 0.025119 5531    3.725    3.823
##  Pilsner   3.690 0.032428 5531    3.627    3.754
##  Porter    3.968 0.023046 5531    3.923    4.014
##  Stout     3.998 0.019908 5531    3.959    4.037
##  Wheat     3.711 0.021229 5531    3.670    3.753
##
## Confidence level used: 0.95
```

Draw a plot that displays, on a single axes, the distribution of the ratings within each category on the same plot

```r
rating.category <- ggplot(summary(emmeans.rating.category),
                      aes(x=category, y=emmean, ymin=lower.CL, ymax=upper.CL)) +
  geom_point() + geom_linerange(color="red") + geom_line(aes(group="category"))+
  labs(x="category",y="rating",subtitle="The mean and 95% CIs for each category")

# Reorder the category in the plot
rating.category.order <- summary(emmeans.rating.category)
## get the order according to emmean
order.category <- rating.category.order[order(rating.category.order$emmean,
                                              decreasing = TRUE),]$category
## reorder
new.rating.category <-  rating.category.order %>% arrange(factor(category,
                                                levels = order.category))

# Violin plot
violin.plot <- ggplot(new.rating.category,
   aes(x=reorder(category, -emmean), y=emmean, ymin=lower.CL, ymax=upper.CL)) +
  geom_point() + geom_linerange(color="red") + geom_line(aes(group="category"))+
  geom_violin(data = new_beer_data,
    aes(x= category, y=rating, ymin=NULL, ymax=NULL, fill=category), alpha=0.5)+
  labs(subtitle = "the distribution of the ratings within each category")
```

Showing whether, on average, a beer receives a higher rating if it has a higher or lower ABV

```r
# Use linear model to observe the relationship of rating and ABV
## NHST approach
lm.rating.ABV <- lm(rating~ABV, data=new_beer_data)
summary(lm.rating.ABV)
```

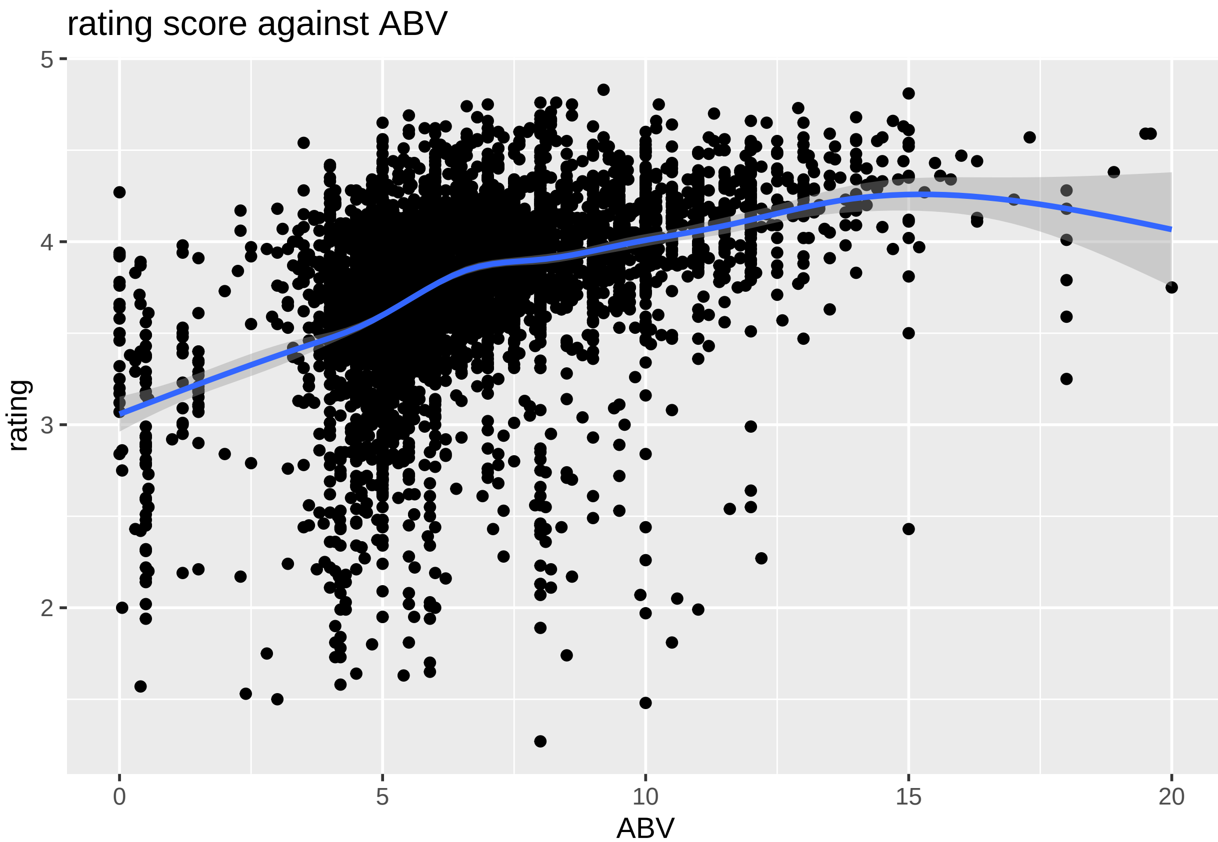
```
##
## Call:
## lm(formula = rating ~ ABV, data = new_beer_data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.60115 -0.15117  0.04883  0.22883  1.03873
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 3.231266   0.015894  203.31   <2e-16 ***
## ABV         0.079985   0.002264   35.34   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.4003 on 5538 degrees of freedom
## Multiple R-squared:  0.184,  Adjusted R-squared:  0.1838
## F-statistic:  1249 on 1 and 5538 DF,  p-value: < 2.2e-16
```

```
## Estimation approach
cbind(coefficient = coef(lm.rating.ABV), confint(lm.rating.ABV),
      Pr = rep("***"))
```

```
##             coefficient          2.5 %                97.5 %               Pr
## (Intercept) "3.2312659539234"    "3.20010818742008"   "3.26242372042673"   "***"
## ABV         "0.0799851335995921" "0.0755477008123751" "0.0844225663868092" "***"
```

```
# Use graph to observe the relationship of rating and ABV
ABV.rating.plot <- ggplot(new_beer_data, aes(x = ABV, y = rating)) +
  geom_point() + geom_smooth() + labs(title = "rating score against ABV")
ABV.rating.plot
```

```
## `geom_smooth()` using method = 'gam' and formula 'y ~ s(x, bs = "cs")'
```




```
# Correlation check
rcorr(as.matrix(select(new_beer_data,rating, ABV), type = "pearson"))
```

```
##         rating  ABV
## rating    1.00 0.43
## ABV       0.43 1.00
##
## n= 5540
##
##
## P
##         rating ABV
## rating          0
## ABV     0
```

Use linear model to show if having more or less Sweet or Malty elements in the flavour results in higher or lower ratings.

```
#NHST approach
## Get main effect linear model
lm.sweet.malty.ABV.rating <- lm(rating~Sweet+Malty+ABV, data=new_beer_data)
summary(lm.sweet.malty.ABV.rating)
```

```
##
## Call:
```

```
## lm(formula = rating ~ Sweet + Malty + ABV, data = new_beer_data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.55353 -0.15606  0.04445  0.22898  1.07441
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 3.2042842  0.0166774 192.133  < 2e-16 ***
## Sweet       0.0014802  0.0001982   7.467 9.48e-14 ***
## Malty       0.0002375  0.0001455   1.632    0.103
## ABV         0.0696706  0.0024886  27.996  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3968 on 5536 degrees of freedom
## Multiple R-squared:  0.1984, Adjusted R-squared:  0.1979
## F-statistic: 456.6 on 3 and 5536 DF,  p-value: < 2.2e-16
```

```
## Get interaction effect model
lm.sweet.malty.ABV.rating.interaction <- lm(rating~Sweet*Malty*ABV,
                                            data=new_beer_data)
summary(lm.sweet.malty.ABV.rating.interaction)
```

```
##
## Call:
## lm(formula = rating ~ Sweet * Malty * ABV, data = new_beer_data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.51815 -0.15624  0.04041  0.22990  1.14263
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)      3.124e+00  3.864e-02  80.848  < 2e-16 ***
## Sweet            6.381e-03  7.637e-04   8.355  < 2e-16 ***
## Malty           -2.719e-03  6.995e-04  -3.887 0.000103 ***
## ABV              8.020e-02  5.752e-03  13.942  < 2e-16 ***
## Sweet:Malty      7.713e-07  9.243e-06   0.083 0.933499
## Sweet:ABV       -6.223e-04  9.932e-05  -6.265 4.01e-10 ***
## Malty:ABV        4.669e-04  9.558e-05   4.886 1.06e-06 ***
## Sweet:Malty:ABV -9.290e-07  1.128e-06  -0.824 0.410135
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.393 on 5532 degrees of freedom
## Multiple R-squared:  0.2142, Adjusted R-squared:  0.2132
## F-statistic: 215.4 on 7 and 5532 DF,  p-value: < 2.2e-16
```

```
## Use ANOVA to check whether adding interaction is helpful to explain the
## dependent variable
anova(lm.sweet.malty.ABV.rating, lm.sweet.malty.ABV.rating.interaction)
```

```
## Analysis of Variance Table
```

```
##
## Model 1: rating ~ Sweet + Malty + ABV
## Model 2: rating ~ Sweet * Malty * ABV
##   Res.Df    RSS Df Sum of Sq      F    Pr(>F)
## 1   5536 871.71
## 2   5532 854.51  4    17.193 27.827 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
better.model <- summary(lm.sweet.malty.ABV.rating.interaction)

# Estimation approach: showing the coefficient and confindence interval of the
## better model in previous step
better.model.pvalue <- better.model$coefficients[,4]
cbind(coefficient = coef(lm.sweet.malty.ABV.rating.interaction),
      confint(lm.sweet.malty.ABV.rating.interaction),
      ifelse(better.model.pvalue <0.01, "***",
      ifelse(better.model.pvalue < 0.05, "**",
      ifelse(better.model.pvalue < 0.1, "*"," "))))
```

```
##                   coefficient             2.5 %                   97.5 %
## (Intercept)       "3.12436537645014"      "3.04860607338532"      "3.20012467951496"      "***"
## Sweet             "0.00638076492277521"   "0.00488352753198744"   "0.00787800231356298"   "***"
## Malty             "-0.00271882971899212"  "-0.00409009074100717"  "-0.00134756869697707"  "***"
## ABV               "0.0801974637011654"    "0.0689211585021404"    "0.0914737689001903"    "***"
## Sweet:Malty       "7.7128286819626e-07"   "-1.73480577976875e-05" "1.88906235340801e-05"  " "
## Sweet:ABV         "-0.000622265769879115" "-0.000816975087719121" "-0.000427556452039109" "***"
## Malty:ABV         "0.000466946752597862"  "0.000279579062628674"  "0.00065431444256705"   "***"
## Sweet:Malty:ABV   "-9.29012153662999e-07" "-3.1399951874275e-06"  "1.2819708801015e-06"   " "
```

```
# Check whether VIF is lower than 5
vif(lm.sweet.malty.ABV.rating)
```

```
##    Sweet    Malty      ABV
## 1.706621 1.450661 1.229980
```

```
vif(lm.sweet.malty.ABV.rating.interaction)
```

```
##           Sweet           Malty             ABV     Sweet:Malty       Sweet:ABV       Malty:ABV
##       25.823000       34.175898        6.698302       65.766333       45.784731       53.219743
## Sweet:Malty:ABV
##       93.598032
```

```
# Apply Pearson correlation to check the statistical relationship
## between variables
rcorr(as.matrix(select(new_beer_data,rating, ABV, Sweet, Malty),
                type = "pearson"))
```

```
##         rating  ABV Sweet Malty
## rating    1.00 0.43  0.29  0.16
## ABV       0.43 1.00  0.43  0.20
```

```
## Sweet    0.29 0.43  1.00  0.56
## Malty    0.16 0.20  0.56  1.00
##
## n= 5540
##
##
## P
##        rating ABV Sweet Malty
## rating         0   0     0
## ABV     0          0     0
## Sweet   0      0         0
## Malty   0      0   0
```

test the effects of flavourings

What flavourings should the company use more/less of if they are creating a high ABV beer? What flavourings should the company use more/less of if they are creating a low ABV beer

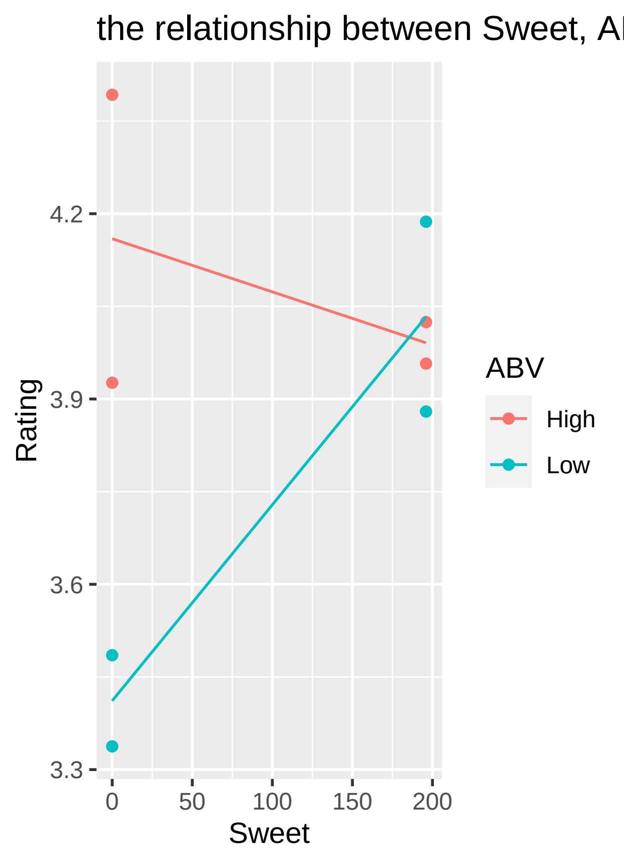
```r
# Define low and high quantiles of ABV values and combine with all situation
## (zero, minimum and maximum Sweet and Malty)
tibble("90th" = quantile(new_beer_data$ABV, 0.9),
"80th" = quantile(new_beer_data$ABV, 0.8),
"20th" = quantile(new_beer_data$ABV, 0.2),
"10th" = quantile(new_beer_data$ABV, 0.1))
```

```
## # A tibble: 1 x 4
##   `90th` `80th` `20th` `10th`
##    <dbl>  <dbl>  <dbl>  <dbl>
## 1     10    8.2      5    4.5
```

```r
# Since the values of 80th of ABV and 20th of ABV are close,
## so we select the 90th and 10th of ABV to see the difference

## Set up a tibble for the prediction of rating score
### high ABV
preds.rating.intr.highABV <-tibble(ABV =rep(quantile(new_beer_data$ABV,0.9), 4),
                                   Sweet = rep(c(max(new_beer_data$Sweet),
                                   Malty = c(rep(min(new_beer_data$Malty),2),
                                             rep(max(new_beer_data$Malty),2)))

### low ABV
preds.rating.intr.lowABV <- tibble(ABV =rep(quantile(new_beer_data$ABV,0.1), 4),
                                   Sweet = rep(c(max(new_beer_data$Sweet),
                                   Malty = c(rep(min(new_beer_data$Malty),2),
                                             rep(max(new_beer_data$Malty),2)))

## Use the better model to predict the rating score with the data in the
## tibbles generated above and add the rating score into the tibble

### high ABV
preds.rating.intr.highABV <- mutate(preds.rating.intr.highABV,
                    Rating.hat = predict(lm.sweet.malty.ABV.rating.interaction,

### low ABV
```

```r
preds.rating.intr.lowABV <- mutate(preds.rating.intr.lowABV,
                    Rating.hat = predict(lm.sweet.malty.ABV.rating.interaction,

### high and low ABV
preds.rating.intr.allABV <- rbind(preds.rating.intr.highABV,
                                  preds.rating.intr.lowABV)
```

```r
preds.rating.intr.allABV.f <- preds.rating.intr.allABV
# Change factor levels
levels(preds.rating.intr.allABV.f$ABV) <- list("High" = 10,
                               "Low" = 4.5)

by.sweet.rating <- preds.rating.intr.allABV.f %>%
  group_by(ABV, Sweet) %>%
  mutate(rating.sweet = mean(Rating.hat))

by.malty.rating <- preds.rating.intr.allABV.f %>%
  group_by(ABV, Malty) %>%
  mutate(rating.malty = mean(Rating.hat))

sweet.ABV.rating.plot <- ggplot(preds.rating.intr.allABV.f) +
  geom_point(aes(x = Sweet, y = Rating.hat,color = ABV)) +
  geom_line(data = by.sweet.rating,
            aes(x=Sweet, y=rating.sweet, group=ABV, colour=ABV)) +
  ylab("Predicted Rating") + labs(x="Sweet", y="Rating",
                    title = "the relationship between Sweet, ABV and rating")

tt <- ttheme_default(colhead=list(fg_params = list(parse=TRUE)))

# Set the legend for the column explanation
FigLegend1 <-data.frame(legend="rating.sweet: It is the mean of the rating under
                        a specific degree of sweetness")
legend.graph.sweet <- tableGrob(FigLegend1, rows = NULL, cols = NULL,
                                theme=ttheme_minimal(base_size = 10))

# Put the table, legend and graph side by side
tbl.sweet <- tableGrob(by.sweet.rating, rows = NULL, theme=ttheme_minimal(base_size = 10))

tbl.sweet.legend <- arrangeGrob(tbl.sweet,legend.graph.sweet,nrow = 2,
                                heights = unit(c(2, 0.25),c("null", "null")))

grid.arrange(sweet.ABV.rating.plot, tbl.sweet.legend,
             nrow=1,ncol=2,
             as.table=TRUE)
```

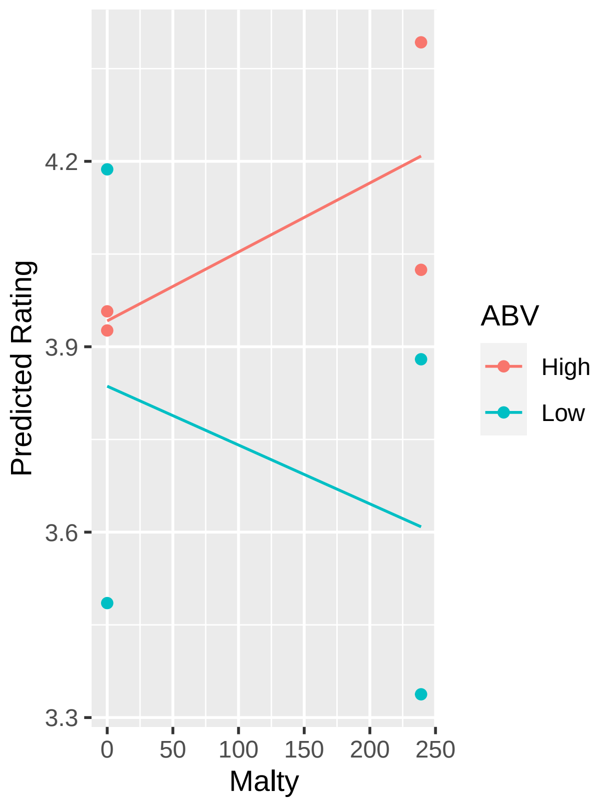


ABV	Sweet	Malty	Rating.hat	rating.sweet
High	196	0	3.957329	3.990902
High	0	0	3.926340	4.159441
High	196	239	4.024475	3.990902
High	0	239	4.392542	4.159441
Low	196	0	4.187045	4.033394
Low	0	0	3.485254	3.411454
Low	196	239	3.879742	4.033394
Low	0	239	3.337655	3.411454

rating.sweet: It is the mean of the rating under
a specific degree of sweetness

```
malty.ABV.rating.plot <- ggplot(preds.rating.intr.allABV.f) +
  geom_point(aes(x = Malty, y = Rating.hat, color = ABV)) +
  geom_line(data = by.malty.rating, aes(x=Malty, y=rating.malty, group=ABV,
          colour=ABV)) + ylab("Predicted Rating") +
  labs(title = "the relationship between Malty, ABV and rating")



# Set the legend for the column explanation
FigLegend2 <-data.frame(legend="rating.malty: It is the mean of the rating under
                         a specific degree of malty")

legend.graph.malty <- tableGrob(FigLegend2, rows = NULL, cols = NULL,
                                theme=ttheme_minimal(base_size = 10))

# Put the table, legend and graph side by side
tbl.malty <- tableGrob(by.malty.rating, rows = NULL, theme = ttheme_minimal(base_size = 10))

tbl.malty.legend <- arrangeGrob(tbl.malty,legend.graph.malty,nrow = 2,
                                heights = unit(c(2, 0.25),c("null", "null")))

grid.arrange(malty.ABV.rating.plot, tbl.malty.legend,
             nrow=1,ncol=2,
             as.table=TRUE)
```

the relationship between Malty, ABV and rating



ABV	Sweet	Malty	Rating.hat	rating.malty
High	196	0	3.957329	3.941835
High	0	0	3.926340	3.941835
High	196	239	4.024475	4.208509
High	0	239	4.392542	4.208509
Low	196	0	4.187045	3.836150
Low	0	0	3.485254	3.836150
Low	196	239	3.879742	3.608699
Low	0	239	3.337655	3.608699

rating.malty: It is the mean of the rating under a specific degree of malty

Section 2 - data analysis and insights

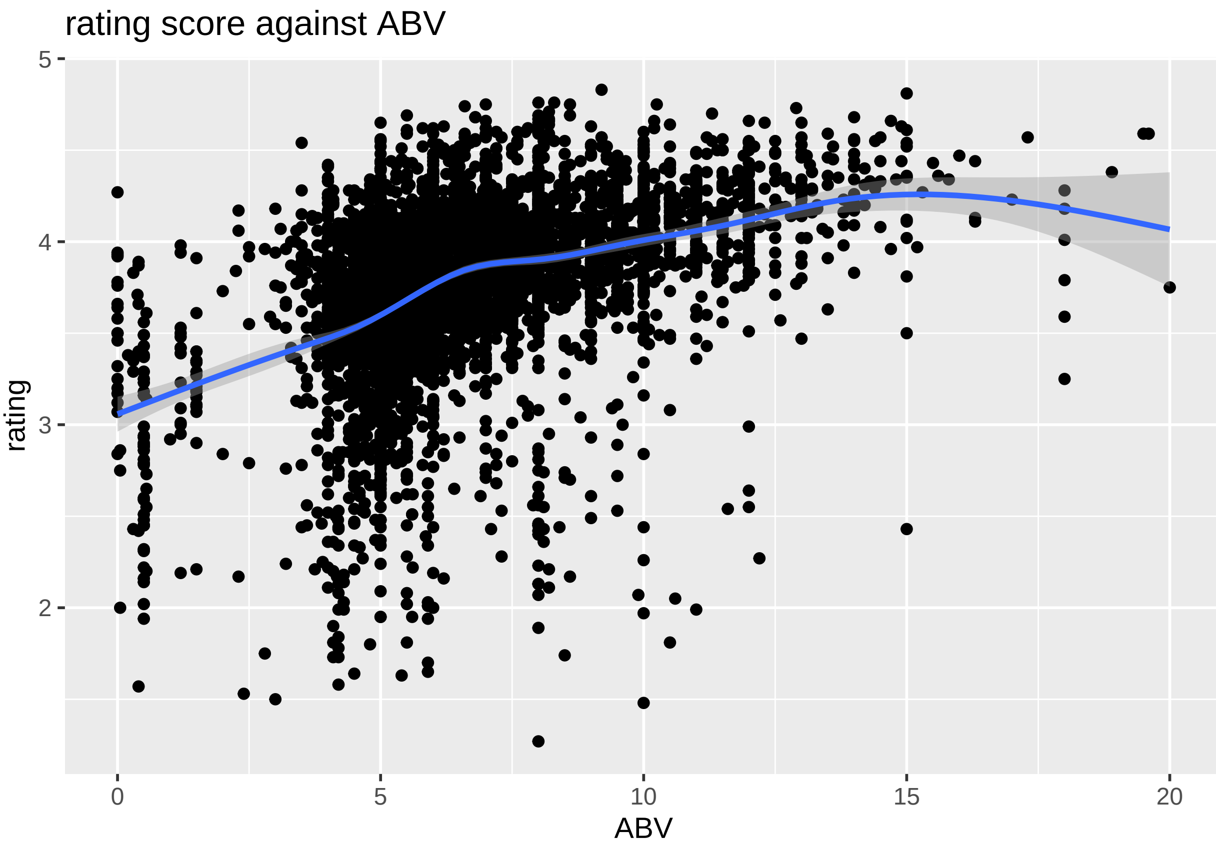
This report presents a data analysis for a beer company. The company want to know whether particular types of beers has higher rating score. In this report, beers are categorised into nine categories (IPA, Lager, Porter, Stout, Wheat, Pale, Pilsner, and Bock) by detecting the strings and identifiers shown in Style column. Afterwards, confidence intervals and means are invested, and the outcomes demonstrated that the means and CIs of rating for each category fall around 3 and 4.

To further the investigation, most of the categories are more concentrated to their means, except for the Lager and Others categories. More specifically, the data of Lager and Others categories are more scattered according to the violin plot. That is, their ratings are various. Also, the mean of data in Lager category has a lower average rating comparing to other categories.

The company also wants to get insights from the data to design a new high-rating product. From the data, we have 5,558 beers and also their properties, including their names, rating score, and multiple flavourings.

According to the graph, linear regression, and correlation test, it can show that rating, ABV, Sweet, and Malty are all positive but low or little correlated. From the correlation test, ABV has higher correlation with rating comparing to other variables. Increasing one unit of ABV would increase the rating by an average of 0.08 points (coefficient = 0.08, 95%CI[0.0755, 0.0844]). Moreover, the p-value of ABV to the rating is significance difference (p-value < 0.05, $t(5538) = 35.34$), so it can be concluded that ABV is also a significant predictor towards the rating.

```
ABV.rating.plot
```



Furthermore, based on the historical data, it can be proved that as the ABV increases, the rating would increase.

Further, based on the model comparison between main effect and interaction effect model (ANOVA test), it can be concluded that the interaction effect model is better because the p-value is less than 0.001, which means the model with the interaction term is significantly improved (F(4, 5532) = 27.827, p-value $<$ 0.001). Thus, the interaction term should be considered when analyze the data.

From the interaction effect model, we can see that Sweet, Malty, and ABV are significant predictors of the model with the p-value lower than 0.001. Moreover, only Malty negatively affects the rating, which means increasing one unit of Malty would decrease 0.003 in the rating. For the Sweet property, increasing one unit of Sweet would lead to an increase of 0.00638 points towars rating(p-value $<$ 0.001, t(5532) = 8.355, coefficient = 0.00638, 95%CI[0.0049, 0.0079]).

For the interaction term only the interaction of Sweet and ABV and the interaction of Malty of ABV are associated with rating. What's more, the interaction of Sweet and ABV is negatively associated to the rating (p-value $<$ 0.001, t(5532) = 6.265, coefficient = -0.0006223, 95%CI[-0.0008, -0.0004]). Although ABV has a positive impact (p-value $<$ 0.001, t(5532) = 13.942, coefficient = 0.08, 95%CI[0.0689, 0.0915]) towards rating, when it interacted with Sweet property, the rating would be negatively impacted. In comparison, the interaction of Malty and ABV positively influence the rating (p-value $<$ 0.001, t(5532) = 4,886, coefficient = 0.0004669, 95%CI[0.0003, 0.0007]).

To investigate how to maximise the rating based on the variables ABV, Sweet, and Malty, the 0.9 and 0.1 quantile of ABV are adopted, and the maximum and minimum of Sweet and Malty are used to predict the rating with the ABV.

According to the graphs and tables about the relationship between ABV, rating, and Sweet/Malty shown above, it can be concluded that to maximise the rating, Sweet should not be added when ABV is high (quantile = 0.9). In contrary, Malty flavours has positive impact towards the rating when ABV is high (quantile = 0.9), so Malty could be added more under high ABV. Also, the rating would increase when Sweet increases or Malty decreases as ABV is low (quantile = 0.1).

According to the findings, if the company are creating a high ABV beer, they should not add Sweet but Malty. On the other hand, for the low ABV beer, the company should add more Sweet and not malty to maximise the rating score. Accordingly, with higher or lower ABVs, the company should execute different plan to design its product.
