Case Study 1

Joaquin

10/8/2021

R Markdown

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --

## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.5     v dplyr   1.0.7
## v tidyr   1.1.3     v stringr 1.4.0
## v readr   2.0.1     v forcats 0.5.1

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(ggpubr)
library(class)
library(caret)
```

```
## Loading required package: lattice

##
## Attaching package: 'caret'

## The following object is masked from 'package:purrr':
##
##     lift
```

```
library(e1071)
beers <- read_csv("https://raw.githubusercontent.com/BivinSadler/MSDS_6306_Doing-Data-Science/Master/Uni
```

```
## Rows: 2410 Columns: 7

## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (2): Name, Style
## dbl (5): Beer_ID, ABV, IBU, Brewery_id, Ounces
```

```
##
## 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
breweries <- read_csv("https://raw.githubusercontent.com/BivinSadler/MSDS_6306_Doing-Data-Science/Maste
```

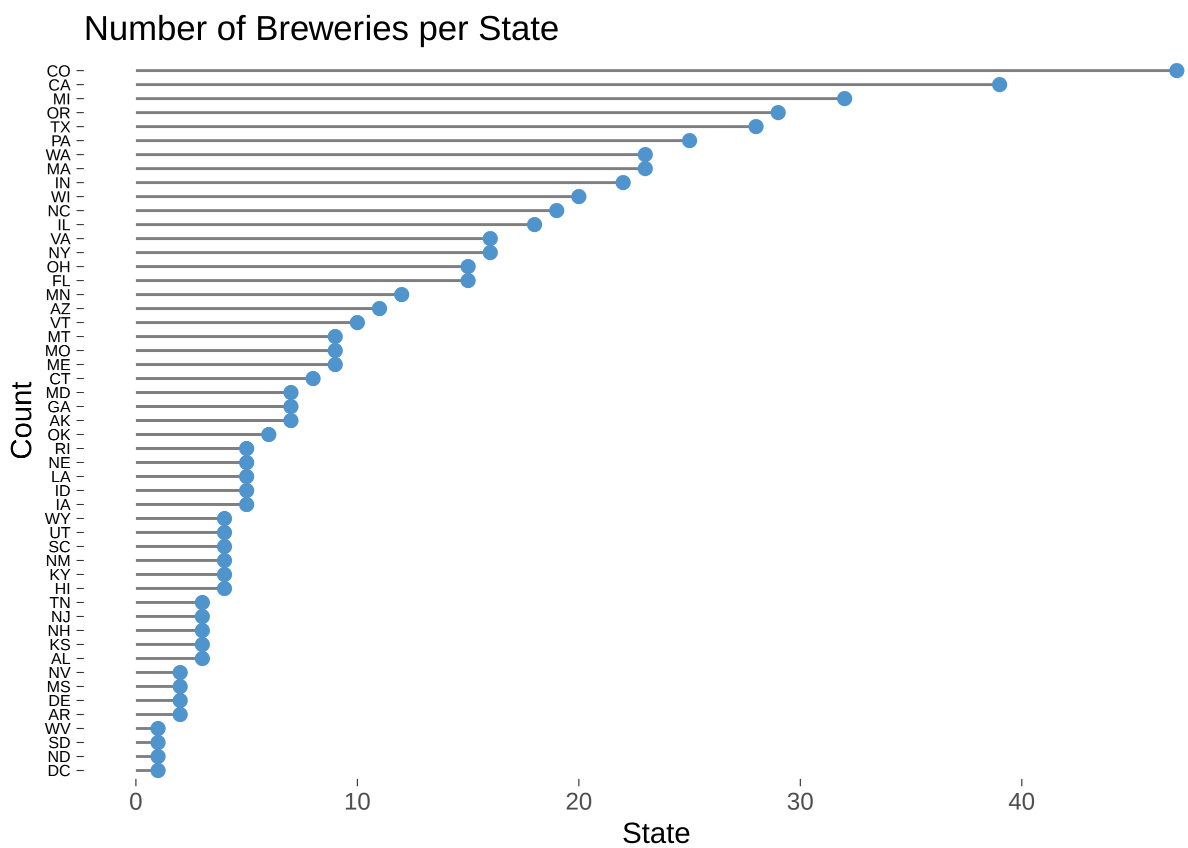
```
## Rows: 558 Columns: 4

## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (3): Name, City, State
## dbl (1): Brew_ID

##
## 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
##How many breweries are present in each state?
#This table outlines the quantity of breweries in each state and will be used as the basis of the follo
numbystate <- data.frame(breweries %>% count(State))
numbystate
```

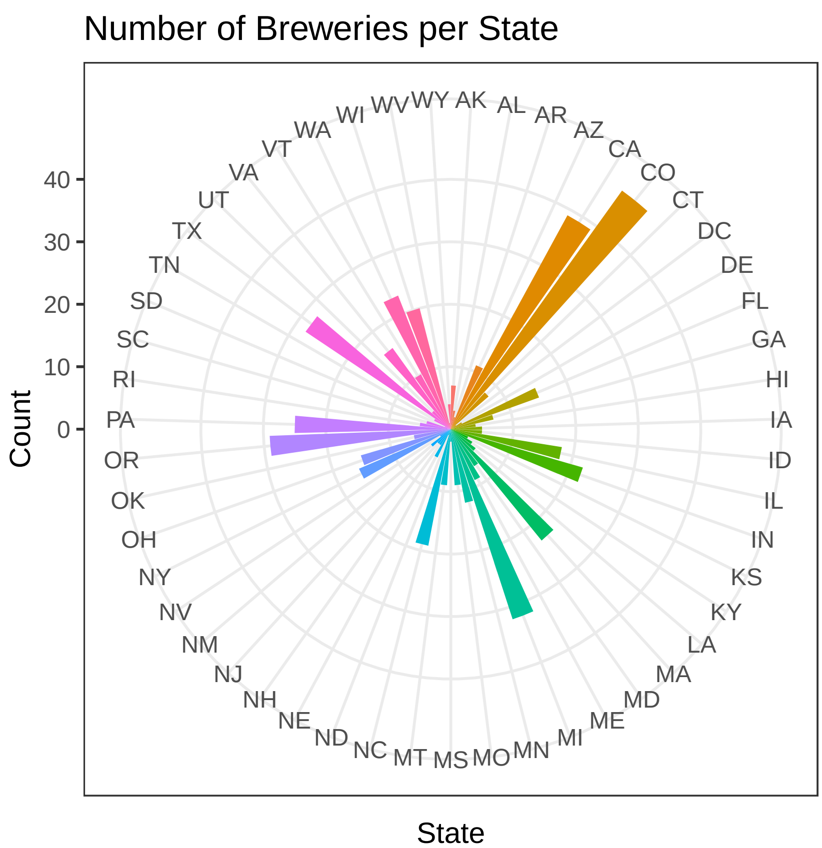
```
##    State  n
## 1     AK  7
## 2     AL  3
## 3     AR  2
## 4     AZ 11
## 5     CA 39
## 6     CO 47
## 7     CT  8
## 8     DC  1
## 9     DE  2
## 10    FL 15
## 11    GA  7
## 12    HI  4
## 13    IA  5
## 14    ID  5
## 15    IL 18
## 16    IN 22
## 17    KS  3
## 18    KY  4
## 19    LA  5
## 20    MA 23
## 21    MD  7
## 22    ME  9
## 23    MI 32
## 24    MN 12
## 25    MO  9
## 26    MS  2
## 27    MT  9
## 28    NC 19
## 29    ND  1
```

```
## 30    NE  5
## 31    NH  3
## 32    NJ  3
## 33    NM  4
## 34    NV  2
## 35    NY 16
## 36    OH 15
## 37    OK  6
## 38    OR 29
## 39    PA 25
## 40    RI  5
## 41    SC  4
## 42    SD  1
## 43    TN  3
## 44    TX 28
## 45    UT  4
## 46    VA 16
## 47    VT 10
## 48    WA 23
## 49    WI 20
## 50    WV  1
## 51    WY  4
```

```
numbystate$State <- as.factor(numbystate$State)

#bar graph depicting the number of breweries in each state
numbystate %>% ggplot(aes(x = reorder(State, n), y = n)) +
  geom_segment(aes(xend=State, yend=0), color = 'grey50') +
  geom_point(size=2, color="steelblue3") +
  coord_flip() +
  theme(legend.position = "none") +
  theme(axis.text.y = element_text(size = 6, color = "black")) +
  theme(panel.background = element_rect(fill = "white")) +
  theme(axis.ticks = element_line(size = .2)) +
  labs(x = "Count", y = "State") +
  ggtitle("Number of Breweries per State")
```




```
#Different chart (polar coordinates) giving a better visualization of relative amount of breweries in e

numbystate %>% ggplot(aes(x = State, y = n, fill = State)) +
  geom_bar(stat = 'identity') +
  coord_polar() + theme_bw() +
  theme(legend.position = "none") +
  labs(x = "State", y = "Count") + ggtitle("Number of Breweries per State")
```

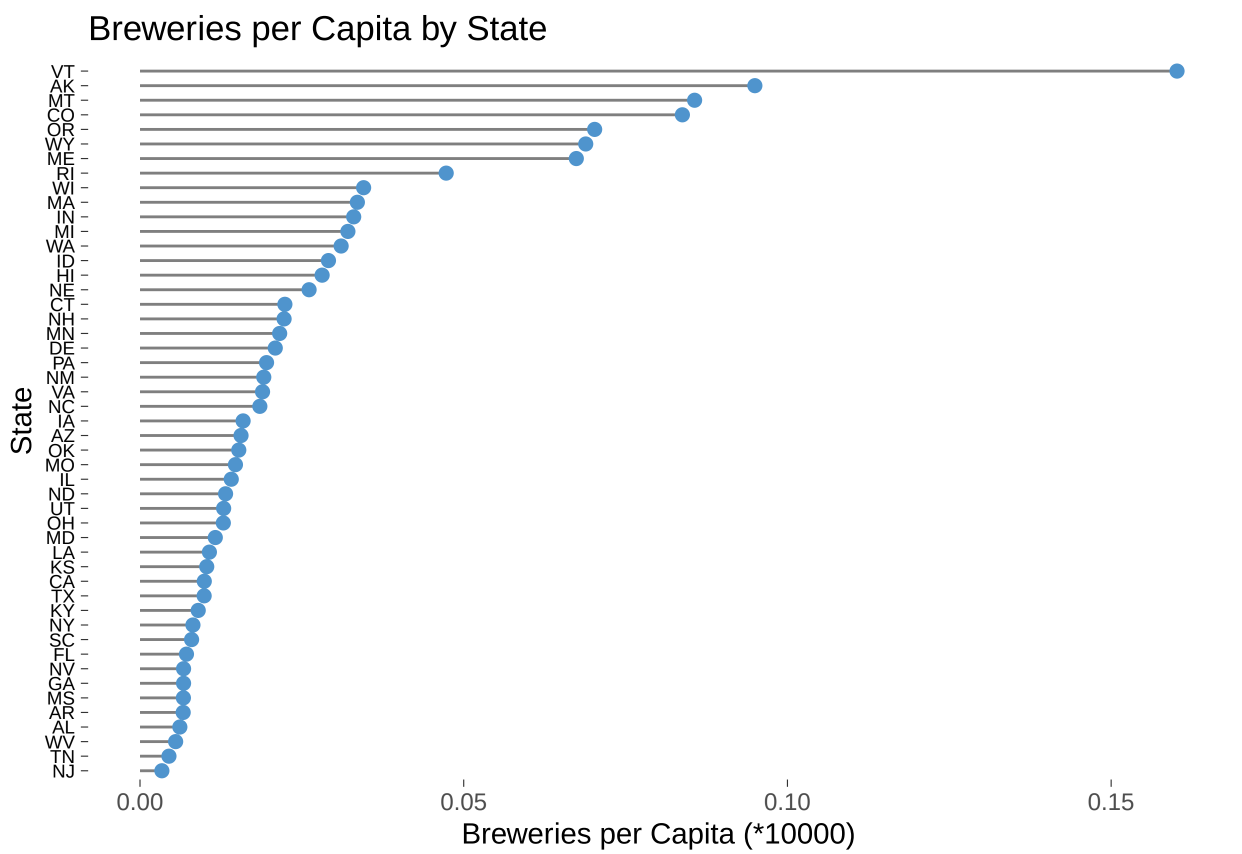



```r
## A more accurate representation of brewery density is depicted by determining Breweries per Capita. Da

statepop <- read_csv("https://raw.githubusercontent.com/j-dominguez9/Case-Study-1/main/statepop.csv")
```

```
## Rows: 50 Columns: 2

## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (1): State
## dbl (1): population

##
## 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
#Join census data with previous table(numbystate)
breweriespercapita <- full_join(numbystate, statepop, by = "State") %>%
  mutate(bpc = (n/population)*10000) %>%
  mutate(State = as.factor(State))

#Plot
breweriespercapita %>% filter(!is.na(population)) %>% filter(!State == "SD") %>%
  ggplot(aes(x = reorder(State, bpc), y = bpc)) +
  geom_segment(aes(xend=State, yend=0), color = 'grey50') +
```

```r
  geom_point(size=2, color="steelblue3") +
  coord_flip() +
  theme(legend.position = "none") +
  theme(axis.text.y = element_text(size = 7, color = "black")) +
  theme(panel.background = element_rect(fill = "white")) +
  theme(axis.ticks = element_line(size = .2)) +
  ggtitle("Breweries per Capita by State") +
  labs(x = "State", y = "Breweries per Capita (*10000)")
```




```r
##Address the missing values in each column.
sum(is.na(breweries))
```

```
## [1] 0
```

```r
#As we can see, the breweries data set holds no missing values, thus no need to eliminate any missing v

sum(is.na(beers))
```

```
## [1] 1072
```

```r
#The beers dataset has 1072, so we must eliminate noted values.

beers_clean <- beers %>% filter(!is.na(Name)) %>% filter(!is.na(Beer_ID)) %>%
  filter(!is.na(ABV)) %>% filter(!is.na(IBU)) %>%
```

```
  filter(!is.na(Brewery_id)) %>% filter(!is.na(Style)) %>%
  filter(!is.na(Ounces))

#Removed 1007 rows due to missing values.
```

```
##Compute the median alcohol content and international bitterness unit for each state. Plot a bar chart

#In order to join the two data sets successfully, we will need to find a primary key and a foreign key

colnames(breweries)
```

```
## [1] "Brew_ID" "Name"    "City"    "State"
```

```
colnames(beers_clean)
```

```
## [1] "Name"       "Beer_ID"    "ABV"        "IBU"        "Brewery_id"
## [6] "Style"      "Ounces"
```

```
#As we can see, Brewery ID would be a good key to join them; however, we must make sure that they have 

breweries$Brewery = breweries$Name
breweries$Brewery_id = breweries$Brew_ID
head(breweries)
```

```
## # A tibble: 6 x 6
##   Brew_ID Name                      City          State Brewery       Brewery_id
##     <dbl> <chr>                     <chr>         <chr> <chr>              <dbl>
## 1       1 NorthGate Brewing         Minneapolis   MN    NorthGate Br~          1
## 2       2 Against the Grain Brewery Louisville    KY    Against the ~          2
## 3       3 Jack's Abby Craft Lagers  Framingham    MA    Jack's Abby ~          3
## 4       4 Mike Hess Brewing Company San Diego     CA    Mike Hess Br~          4
## 5       5 Fort Point Beer Company   San Francisco CA    Fort Point B~          5
## 6       6 COAST Brewing Company     Charleston    SC    COAST Brewin~          6
```

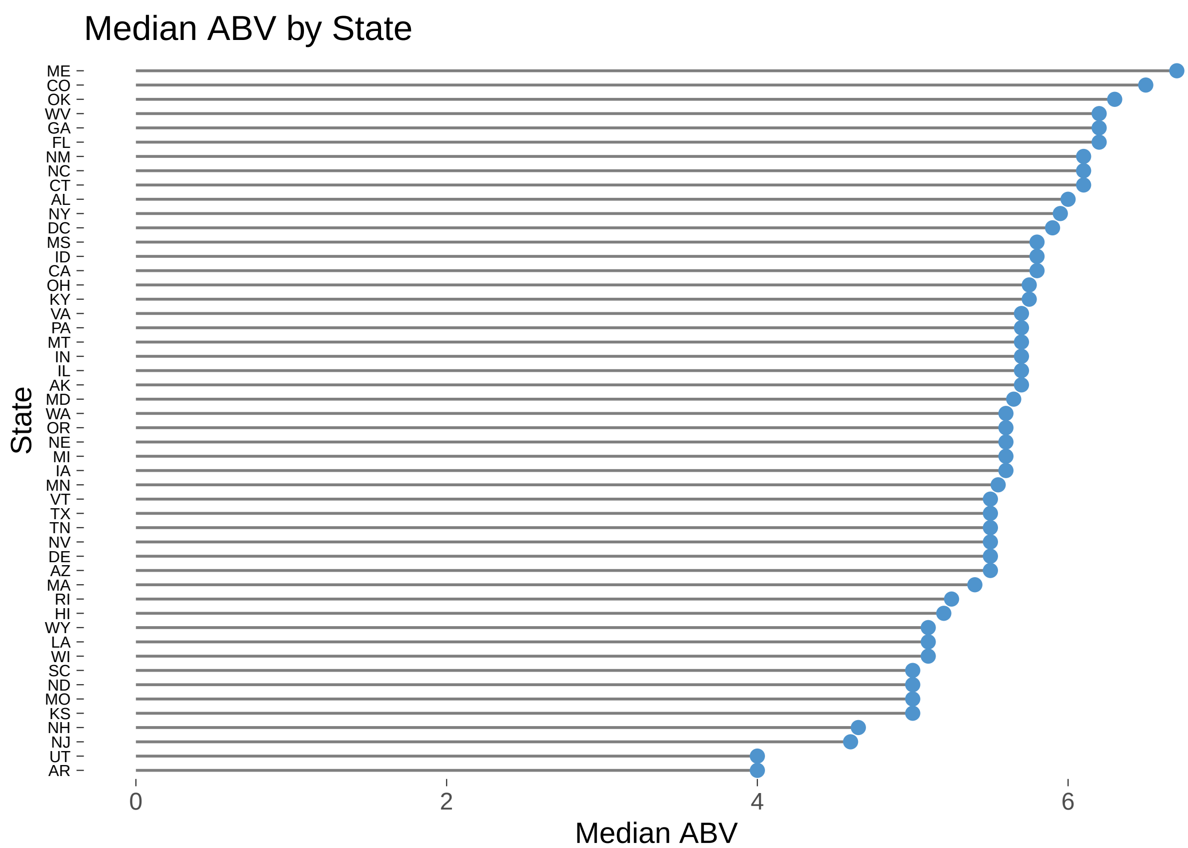
```
#Let's create a new dataframe with the relevant columns from 'breweries' before joining.

breweries_clean <- breweries %>% select(Brewery, City, State, Brewery_id)
join <- inner_join(breweries_clean, beers_clean, by = "Brewery_id")

#From the joined data frame, let's select the relevant columns.

medABVIBU <- join %>% select(State, ABV, IBU)
head(medABVIBU)
```

```
## # A tibble: 6 x 3
##   State   ABV   IBU
##   <chr> <dbl> <dbl>
## 1 MN    0.045    50
## 2 MN    0.049    26
## 3 MN    0.048    19
## 4 MN    0.06     38
## 5 MN    0.06     25
## 6 MN    0.056    47
```

```r
#Having the relevant columns to work with, lets create a new one with the median ABV of each state and
medABV <- medABVIBU %>% group_by(State) %>% mutate(medianABV = median(ABV)*100) %>% select(State, mediar
head(medABV)
```

```
## # A tibble: 6 x 2
## # Groups:   State [6]
##   State medianABV
##   <chr>     <dbl>
## 1 AK          5.7
## 2 AL          6
## 3 AR          4
## 4 AZ          5.5
## 5 CA          5.8
## 6 CO          6.5
```

```r
#As you can see, there is a row for every beer produced in each state. In order to plot the median corr



##Data set is now ready to plot.

medABV %>% ggplot(aes(x = reorder(State, medianABV), y = medianABV)) +
geom_segment(aes(xend=State, yend=0), color = 'grey50') +
  geom_point(size=2, color="steelblue3") +
  coord_flip() +
  theme(legend.position = "none") +
  theme(axis.text.y = element_text(size = 6, color = "black")) +
  theme(panel.background = element_rect(fill = "white")) +
  theme(axis.ticks = element_line(size = .2)) +
  labs(x = "State", y = "Median ABV") +
  ggtitle("Median ABV by State")
```

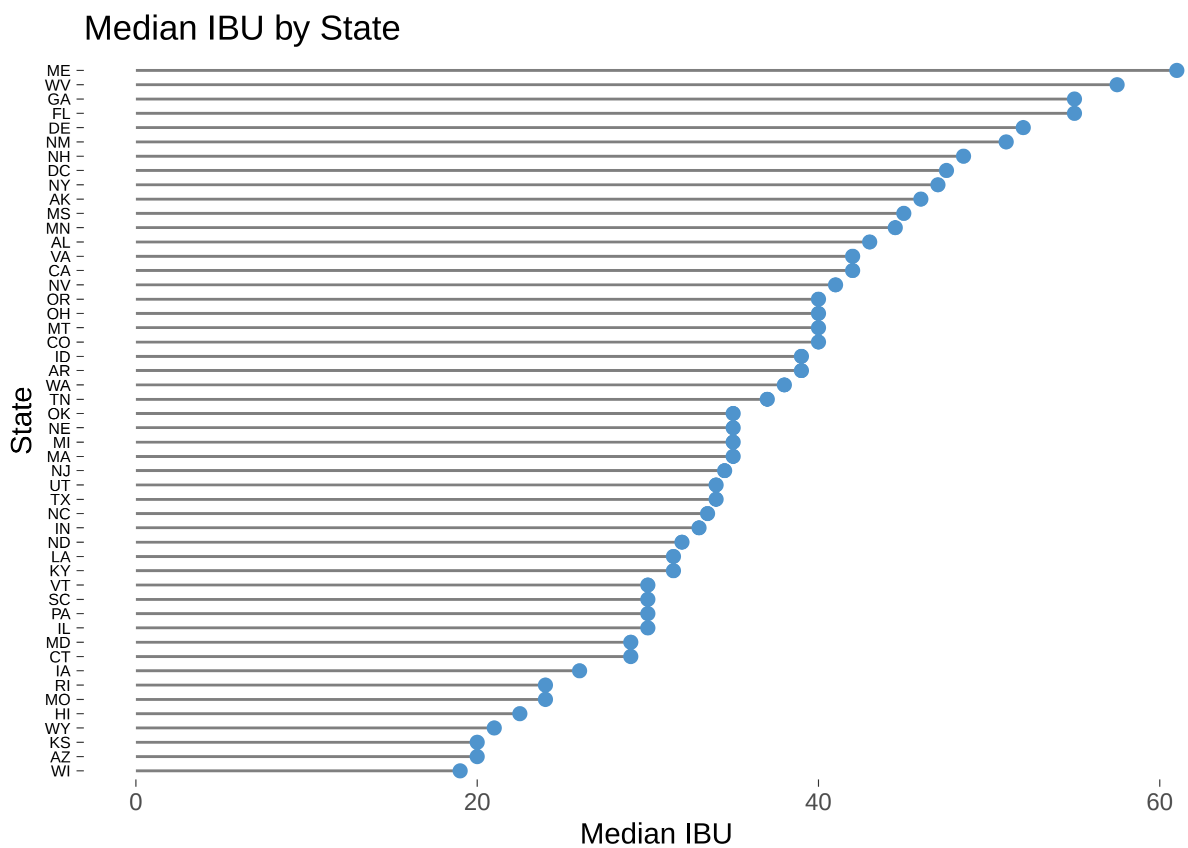



```
##We will follow a similar process to derive median IBU by State.

medIBU <- medABVIBU %>%
  group_by(State) %>%
  mutate(medianIBU = median(IBU)) %>%
  select(State, medianIBU) %>%
  arrange(State) %>%
  distinct(State, medianIBU)

head(medIBU)
```

```
## # A tibble: 6 x 2
## # Groups:   State [6]
##   State medianIBU
##   <chr>     <dbl>
## 1 AK           46
## 2 AL           43
## 3 AR           39
## 4 AZ           20
## 5 CA           42
## 6 CO           40
```

```
#And plot in a similar manner.

medIBU %>% ggplot(aes(x = reorder(State, medianIBU), y = medianIBU)) +
```

```r
 geom_segment(aes(xend=State, yend=0), color = 'grey50') +
  geom_point(size=2, color="steelblue3") +
  coord_flip() +
  theme(legend.position = "none") +
  theme(axis.text.y = element_text(size = 6, color = "black")) +
  theme(panel.background = element_rect(fill = "white")) +
  theme(axis.ticks = element_line(size = .2)) +
  labs(x = "State", y = "Median IBU") +
  ggtitle("Median IBU by State")
```




```r
##Which state has the maximum alcoholic (ABV) beer? Which state has the most bitter (IBU) beer?

#We will create a new data frame with relevant columns to explore this question.
maxABVIBU <- join %>% select(Brewery, Name, State, ABV, IBU)

#We need to identify the highest ABV level
max(maxABVIBU$ABV)
```

```
## [1] 0.125
```

```r
maxABVIBU %>% filter(ABV == "0.125")
```

```
## # A tibble: 1 x 5
##   Brewery                    Name            State   ABV   IBU
```

```
##   <chr>                     <chr>          <chr> <dbl> <dbl>
## 1 Against the Grain Brewery London Balling KY    0.125    80
```

```
## We can see that the beer with the highest ABV belongs to the state of Kentucky (KY) with an ABV of 1

#We follow the same process for IBU.
max(maxABVIBU$IBU)
```

```
## [1] 138
```

```
maxABVIBU %>% filter(IBU == "138")
```

```
## # A tibble: 1 x 5
##   Brewery                 Name                       State   ABV   IBU
##   <chr>                   <chr>                      <chr> <dbl> <dbl>
## 1 Astoria Brewing Company Bitter Bitch Imperial IPA OR    0.082   138
```

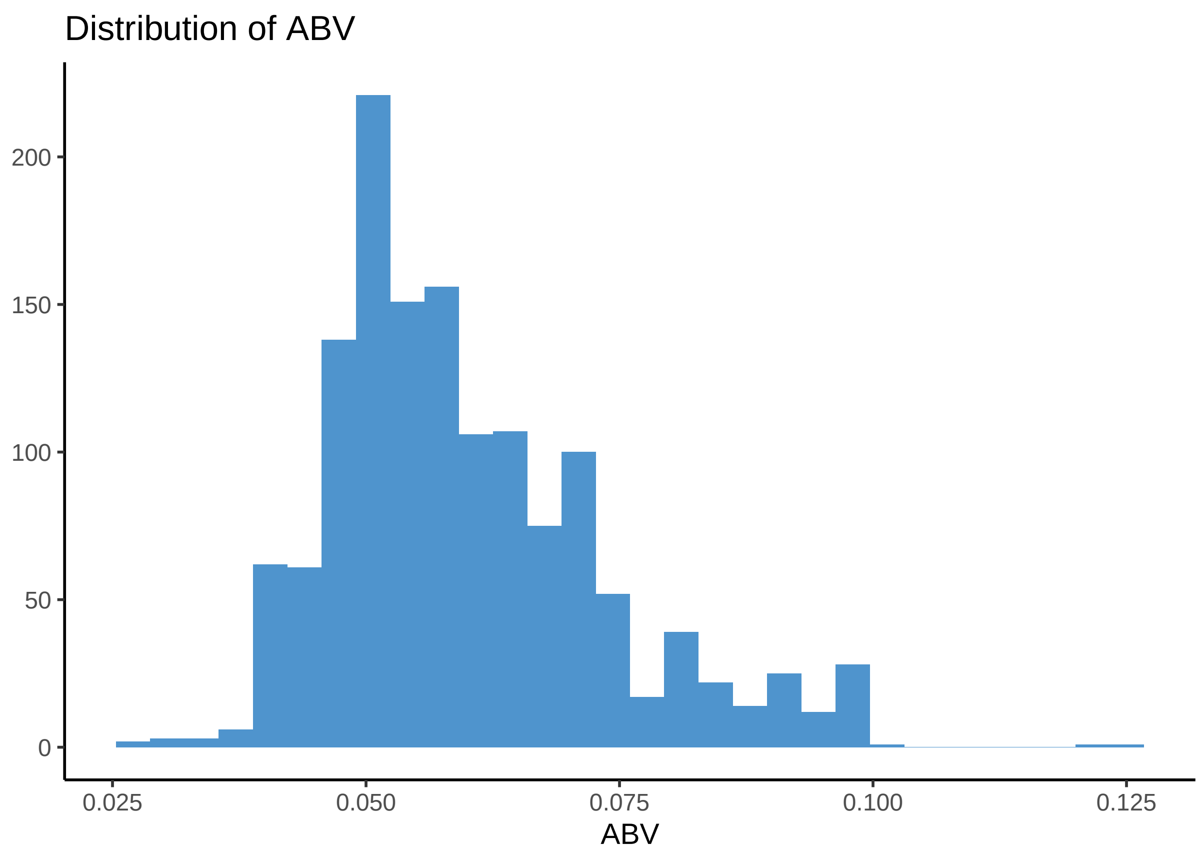
```
#The beer with the highest IBU belongs to the state of Oregon (OR) with an IBU of 138. The beer is "Bit
```

```
##Comment on the summary statistics and distribution of the ABV variable.
summary(maxABVIBU$ABV)
```

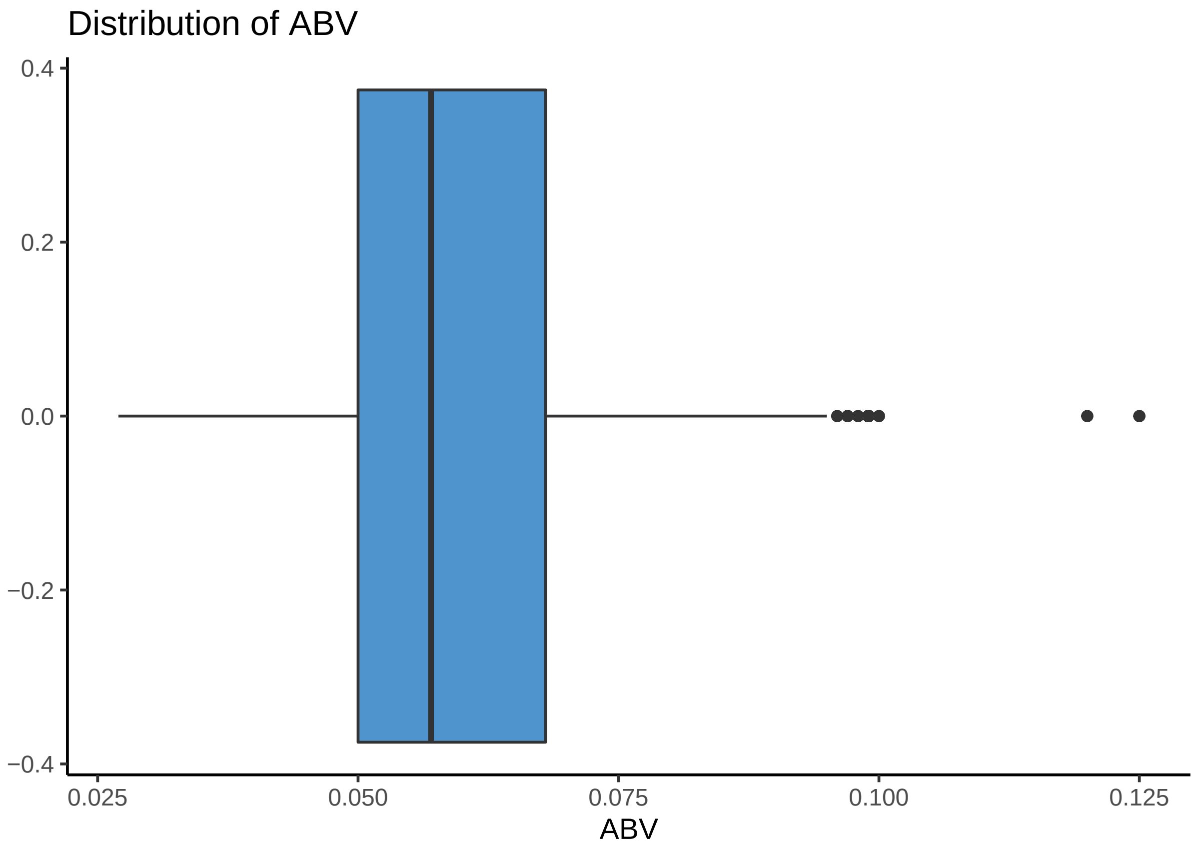
```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## 0.02700 0.05000 0.05700 0.05992 0.06800 0.12500
```

```
#Histogram
maxABVIBU %>% ggplot(aes(x = ABV)) +
  geom_histogram(fill = "steelblue3") +
  theme_classic() +
  labs(x = "ABV", y = "") +
  ggtitle("Distribution of ABV")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

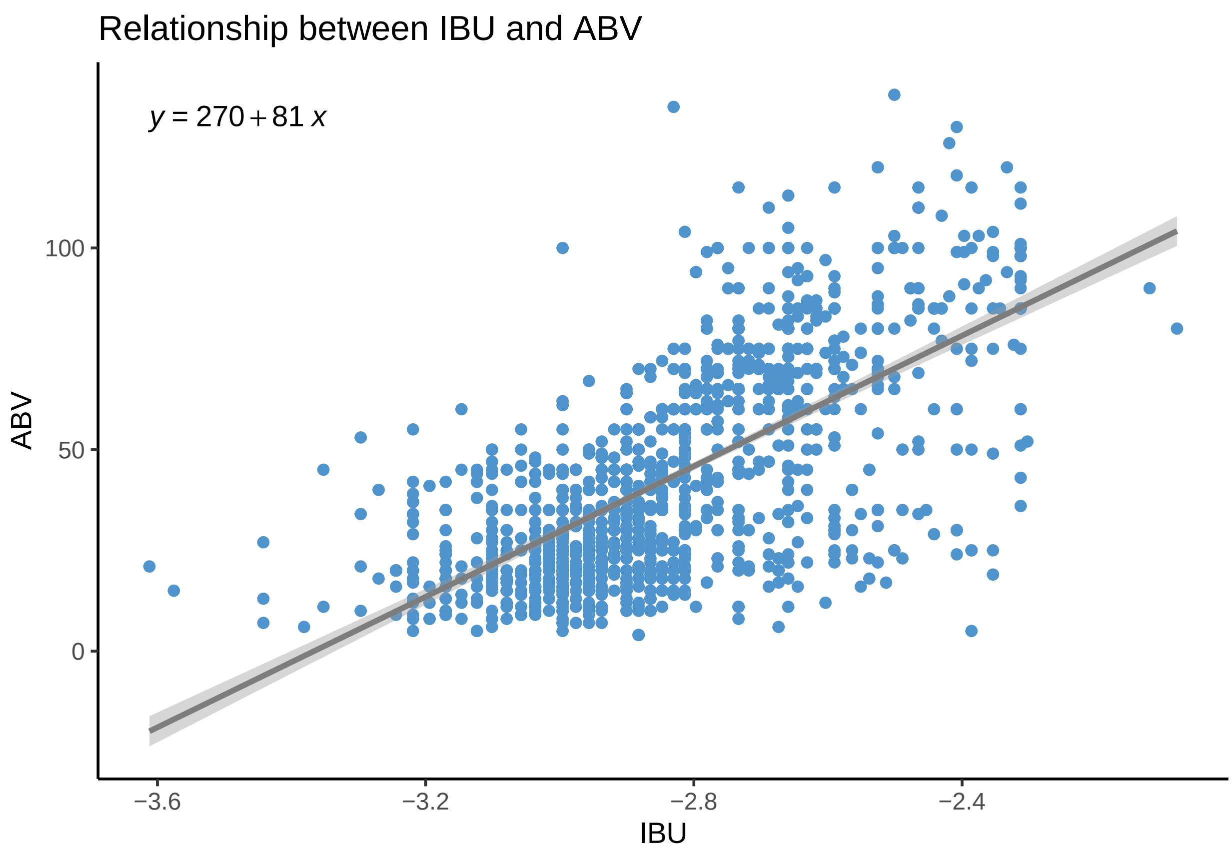



```
#Boxplot
maxABVIBU %>% ggplot(aes(x = ABV)) +
  geom_boxplot(fill = "steelblue3") +
  theme_classic() +
  labs(x = "ABV", y = "") +
  ggtitle("Distribution of ABV")
```




```
##Is there an apparent relationship between the bitterness of the beer and its alcoholic content? Draw

#As we saw earlier, the distributions of both the IBU and ABV columns were right skewed, it'd be helpfu

maxABVIBU %>% ggplot(aes(x = log(ABV), y = IBU)) +
  geom_point(color = "steelblue3") +
  geom_smooth(method = "lm", color = "grey49") +
  stat_regline_equation() +
  theme_classic() +
  labs(x = "IBU", y = "ABV") +
  ggtitle("Relationship between IBU and ABV")
```

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




```
##From a visual inspection as well as a simple linear regression model, we can say that there is an appa

cor.test(x = log(maxABVIBU$ABV), y = maxABVIBU$IBU)
```

```
##
##  Pearson's product-moment correlation
##
## data:  log(maxABVIBU$ABV) and maxABVIBU$IBU
## t = 34.032, df = 1401, p-value < 2.2e-16
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.6430317 0.7004025
## sample estimates:
##       cor
## 0.6727271
```

```
#This Pearson's correlation provides overwhelming evidence that there is a positive linear relationship
```

```
##Budweiser would also like to investigate the difference with respect to IBU and ABV between IPAs (Ind

sum(grepl("Ale", join$Style))
```

```
## [1] 559
```

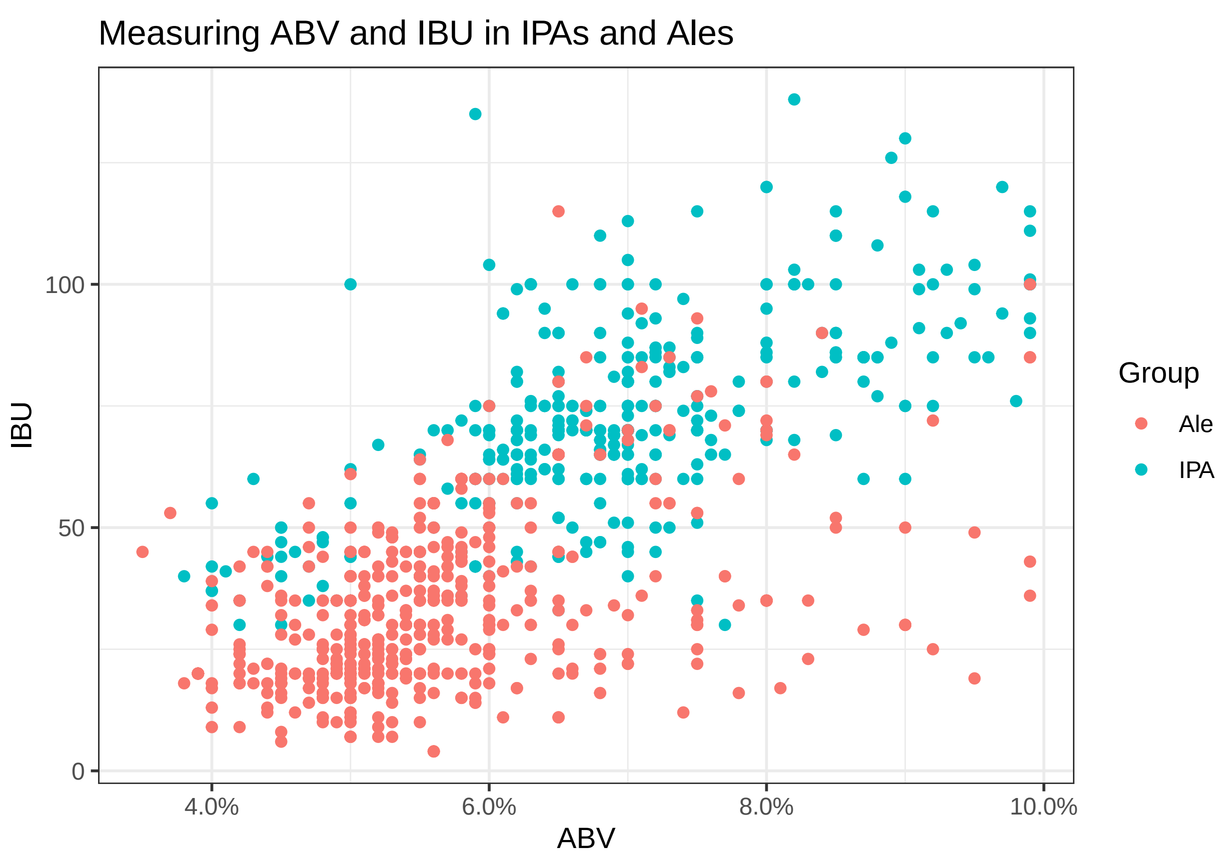
```r
sum(grepl("(India | IPA)", join$Style))
```

```
## [1] 395
```

```r
IPAs <- join %>% filter(grepl("(India | IPA)", Style)) %>% filter(!grepl("Lager", Style))
Ales <- join %>% filter(grepl("Ale", Style)) %>% filter(!grepl("(India | IPA)", Style))

x <- data.frame(Group = "Ale", c(1:552)) %>% select(Group)
final_Ales <- cbind(x, Ales)

y <- data.frame(Group = "IPA", c(1:392)) %>% select(Group)
final_IPAs <- cbind(y, IPAs)

IPA_Ales <- full_join(final_IPAs, final_Ales)
```

```
## Joining, by = c("Group", "Brewery", "City", "State", "Brewery_id", "Name", "Beer_ID", "ABV", "IBU",
```

```r
IPA_Ales %>% ggplot(aes(x = ABV, y = IBU, color = Group)) +
  geom_point() +
  scale_x_continuous(labels = scales::percent) +
  theme_bw() +
  ggtitle("Measuring ABV and IBU in IPAs and Ales")
```

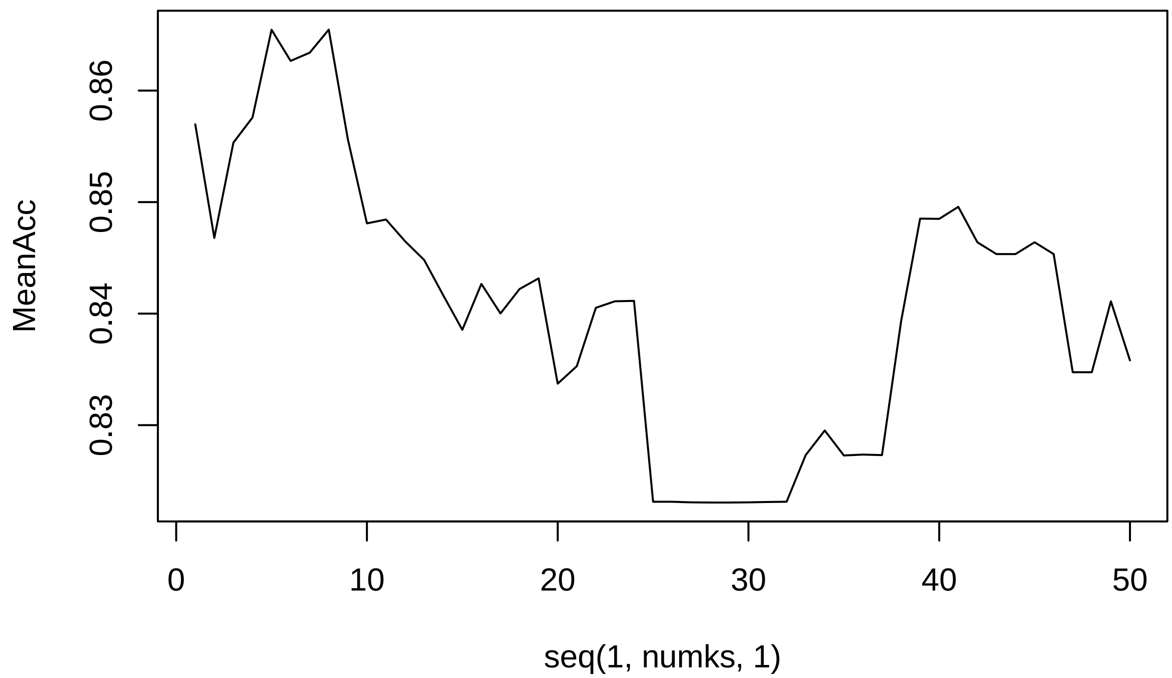


```r
## k-NN model. We first run a simple model to determine the accuracy of the model with k = 3.

confusionMatrix(table(knn.cv(IPA_Ales[,8:9], IPA_Ales$Group, k = 3), IPA_Ales$Group))
```

```
## Confusion Matrix and Statistics
##
##
##       Ale IPA
##   Ale 484  70
##   IPA  68 322
##
##                Accuracy : 0.8538
##                  95% CI : (0.8296, 0.8757)
##     No Information Rate : 0.5847
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.6988
##
##  Mcnemar's Test P-Value : 0.9322
##
##             Sensitivity : 0.8768
##             Specificity : 0.8214
##          Pos Pred Value : 0.8736
##          Neg Pred Value : 0.8256
##              Prevalence : 0.5847
##          Detection Rate : 0.5127
##    Detection Prevalence : 0.5869
##       Balanced Accuracy : 0.8491
##
##        'Positive' Class : Ale
##
```

```r
##We must establish the optimal number for k, with regards to accuracy.

set.seed(1)
iterations = 500
numks = 50

masterAcc = matrix(nrow = iterations, ncol = numks)

for(j in 1:iterations)
{

  for(i in 1:numks)
  {
    CM = confusionMatrix(table(IPA_Ales$Group, knn.cv(IPA_Ales[,8:9], IPA_Ales$Group, k = i)))
    masterAcc[j,i] = CM$overall[1]

  }

}

MeanAcc = colMeans(masterAcc)
```

```r
plot(seq(1,numks,1),MeanAcc, type = "l")
```




```r
which.max(MeanAcc)
```

```
## [1] 8
```

```r
max(MeanAcc)
```

```
## [1] 0.8654661
```

```r
confusionMatrix(table(IPA_Ales$Group, knn.cv(IPA_Ales[,8:9], IPA_Ales$Group, k = 8)))
```

```
## Confusion Matrix and Statistics
##
##
##        Ale IPA
##   Ale 498  54
##   IPA  71 321
##
##                Accuracy : 0.8676
##                  95% CI : (0.8443, 0.8886)
##     No Information Rate : 0.6028
##     P-Value [Acc > NIR] : <2e-16
```

```
##
##                   Kappa : 0.7256
##
##  Mcnemar's Test P-Value : 0.1524
##
##             Sensitivity : 0.8752
##             Specificity : 0.8560
##          Pos Pred Value : 0.9022
##          Neg Pred Value : 0.8189
##              Prevalence : 0.6028
##          Detection Rate : 0.5275
##    Detection Prevalence : 0.5847
##       Balanced Accuracy : 0.8656
##
##        'Positive' Class : Ale
##
```

```
#Naive Bayes
nbIPA_Ales <- IPA_Ales %>% select(Group, ABV, IBU)
naiveBayes(Group~., data = nbIPA_Ales )
```

```
##
## Naive Bayes Classifier for Discrete Predictors
##
## Call:
## naiveBayes.default(x = X, y = Y, laplace = laplace)
##
## A-priori probabilities:
## Y
##       Ale       IPA
## 0.5847458 0.4152542
##
## Conditional probabilities:
##      ABV
## Y            [,1]       [,2]
##   Ale 0.05655616 0.01112430
##   IPA 0.06914286 0.01216069
##
##      IBU
## Y          [,1]     [,2]
##   Ale 34.33333 17.97471
##   IPA 71.94898 19.54567
```

```
iterations = 100
masterAcc = matrix(nrow = iterations)
splitPerc = .8 #Training / Test split Percentage
for(j in 1:iterations)
{
  trainIndices = sample(1:dim(nbIPA_Ales)[1],round(splitPerc * dim(nbIPA_Ales)[1]))
  train = nbIPA_Ales[trainIndices,]
  test = nbIPA_Ales[-trainIndices,]
  model = naiveBayes(train[,2:3],train$Group)
  table(test$Group,predict(model,test[,2:3]))
```

```r
  CM = confusionMatrix(table(test$Group,predict(model,test[,2:3])))
  masterAcc[j] = CM$overall[1]
}
MeanAcc = colMeans(masterAcc)
MeanAcc
```

```
## [1] 0.8419577
```

```r
### t-test
Ales_ABV <- final_Ales %>% select(ABV, Group)
IPAs_ABV <- final_IPAs %>% filter(!grepl("Lager", Style)) %>% select(ABV, Group)

ABVjoin <- full_join(Ales_ABV, IPAs_ABV)
```

```
## Joining, by = c("ABV", "Group")
```

```r
t.test(log(ABV) ~ Group, data = ABVjoin)
```

```
##
##  Welch Two Sample t-test
##
## data:  log(ABV) by Group
## t = -16.91, df = 849.95, p-value < 2.2e-16
## alternative hypothesis: true difference in means between group Ale and group IPA is not equal to 0
## 95 percent confidence interval:
##  -0.2260548 -0.1790364
## sample estimates:
## mean in group Ale mean in group IPA
##         -2.889941         -2.687395
```

```r
t.test(IBU ~ Group, data = IPA_Ales)
```

```
##
##  Welch Two Sample t-test
##
## data:  IBU by Group
## t = -30.118, df = 797.55, p-value < 2.2e-16
## alternative hypothesis: true difference in means between group Ale and group IPA is not equal to 0
## 95 percent confidence interval:
##  -40.06727 -35.16402
## sample estimates:
## mean in group Ale mean in group IPA
##          34.33333          71.94898
```

```r
##NaiveBayes State, Style (IPA, Ales)

nb2 <- IPA_Ales %>% select(State, Group) %>% mutate(State = as.factor(State))
model <- naiveBayes(Group~., data = nb2)
predict(model, data.frame(State = "NH"), type = 'raw')
```

```
##            Ale       IPA
## [1,] 0.3556701 0.6443299
```

```r
###NB State, City, Style (all). Find most popular Style by City
df <- join %>% select(Style, City, State)
model <- naiveBayes(Style~., data = df)
predict(model, data.frame(State = "CO", City = "Buena Vista"))
```

```
## [1] American IPA
## 90 Levels: Abbey Single Ale Altbier ... Witbier
```

```r
###NB Find most popular Style by State
df2 <- df %>% select(Style, State)
model <- naiveBayes(Style~State, data = df2)
predict(model, data.frame(State = "CO"))
```

```
## [1] American Pale Ale (APA)
## 90 Levels: Abbey Single Ale Altbier ... Witbier
```

```r
### NB Find most popular City for Style

df1 <- df %>% select(Style, City)
model <- naiveBayes(City~Style, data = df1)
pred <- predict(model, df1$Style)
CM <- confusionMatrix(as.factor(pred), as.factor(df1$City))
predict(model, data.frame(Style = "American IPA"))
```

```
## [1] San Diego
## 281 Levels: Abingdon Abita Springs Afton Albuquerque Anchorage ... York
```

```r
###NB find most popular State for style

model <- naiveBayes(State~Style, data = df2)
pred <- predict(model, df2$Style)
CM <- confusionMatrix(as.factor(pred), as.factor(df2$State))
predict(model, data.frame(Style = "American IPA"))
```

```
## [1] CA
## 50 Levels: AK AL AR AZ CA CO CT DC DE FL GA HI IA ID IL IN KS KY LA MA ... WY
```