# DS 6306 Project 1

William Jones & Christian Orji

2023-02-19

## Project 1 for DS 6306

## We are tasked with conducting a EDA for the Beers and Breweries Data Set that contains 2410 craft beers and 558 Breweries respectively.

## Data documents

## importing the datasets and analyzing the data types

```
beers = read.csv("D:/Github/MSDS_6306_Doing-Data-Science/Unit 8 and 9 Case Study 1/Beers.csv")
breweries = read.csv("D:/Github/MSDS_6306_Doing-Data-Science/Unit 8 and 9 Case Study 1/Breweries.csv")
summary(beers)
```

```
##      Name              Beer_ID           ABV               IBU
##  Length:2410        Min.   :   1.0   Min.   :0.00100   Min.   :  4.00
##  Class :character   1st Qu.: 808.2   1st Qu.:0.05000   1st Qu.: 21.00
##  Mode  :character   Median :1453.5   Median :0.05600   Median : 35.00
##                     Mean   :1431.1   Mean   :0.05977   Mean   : 42.71
##                     3rd Qu.:2075.8   3rd Qu.:0.06700   3rd Qu.: 64.00
##                     Max.   :2692.0   Max.   :0.12800   Max.   :138.00
##                                      NA's   :62        NA's   :1005
##    Brewery_id       Style               Ounces
##  Min.   :  1.0   Length:2410        Min.   : 8.40
##  1st Qu.: 94.0   Class :character   1st Qu.:12.00
##  Median :206.0   Mode  :character   Median :12.00
##  Mean   :232.7                      Mean   :13.59
##  3rd Qu.:367.0                      3rd Qu.:16.00
##  Max.   :558.0                      Max.   :32.00
##
```

```
summary(breweries)
```

```
##     Brew_ID          Name               City              State
##  Min.   :  1.0   Length:558         Length:558         Length:558
##  1st Qu.:140.2   Class :character   Class :character   Class :character
##  Median :279.5   Mode  :character   Mode  :character   Mode  :character
##  Mean   :279.5
##  3rd Qu.:418.8
##  Max.   :558.0
```

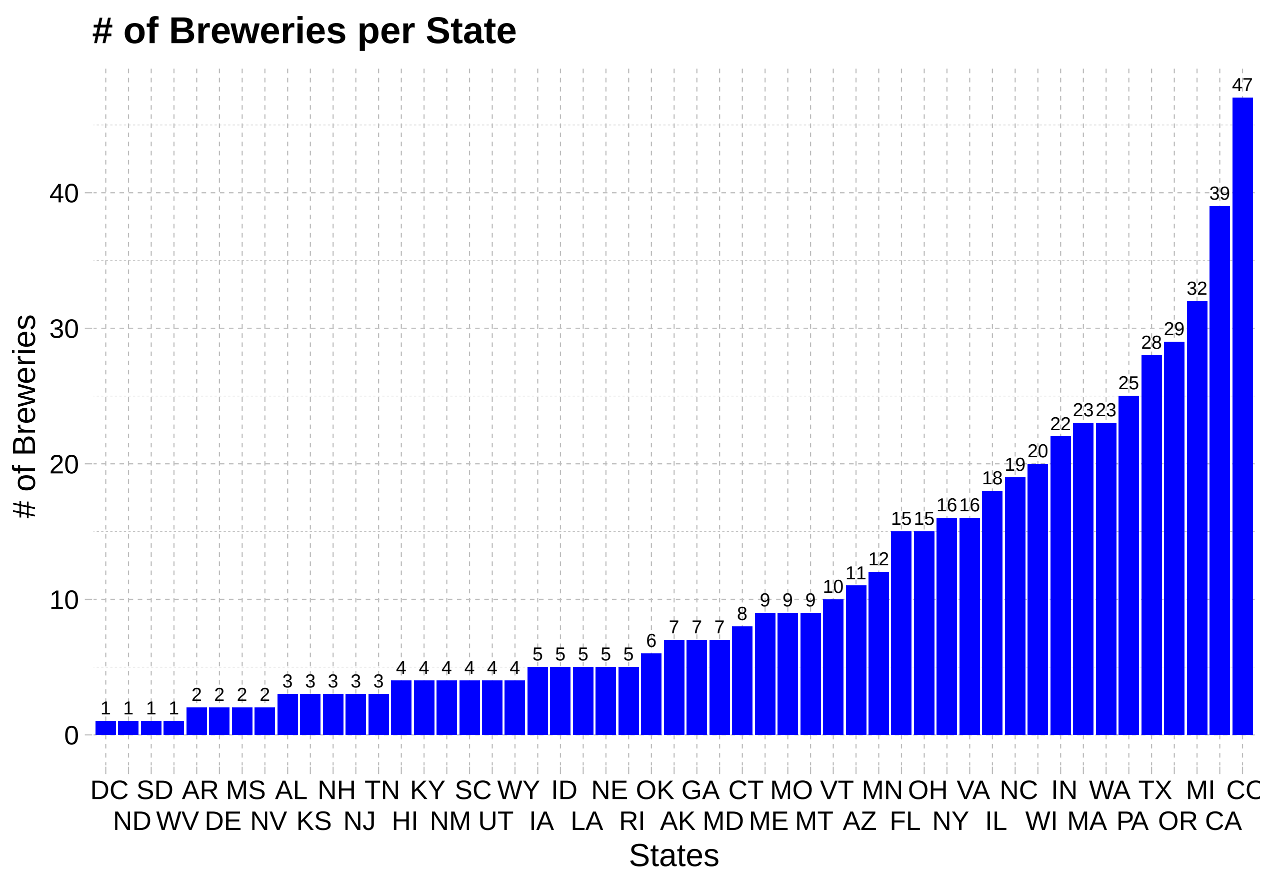
## Question 1

**how many breweries are present in each state?**

```r
library(ggplot2)
library(ggthemes)
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.2 --
## v tibble  3.1.8      v dplyr   1.0.10
## v tidyr   1.2.1      v stringr 1.5.0
## v readr   2.1.3      v forcats 0.5.2
## v purrr   1.0.0
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```r
breweries %>% group_by(State) %>%
  summarise(count = n_distinct(Brew_ID)) %>%
  arrange(count) %>%
  ggplot(aes(x= reorder(State, +count), y = count)) +
  geom_bar(stat="identity", fill ="blue") +
  scale_x_discrete(guide = guide_axis(n.dodge=2)) +
  geom_text(aes(label=count), vjust=-0.5, color="black",
            position = position_dodge(0.9), size=2.5) +
  ylab('# of Breweries') +
  xlab('States') +
  ggtitle('# of Breweries per State') +
  theme_pander()
```



The output is a bar chart showing the number of breweries in each state of the US. The x-axis shows the states and is ordered in ascending order of the number of breweries, while the y-axis shows the count of breweries. The chart is colored in blue and has labels above each bar indicating the count of breweries. Overall, the chart is well-designed and easy to read, and it effectively communicates the information about the distribution of breweries across different states. The use of dodging for the labels and scaling the x-axis helps to prevent overlap and ensure that the chart is easy to read.

## Question 2

**Merging the beer and brewery datasets, both share the Brew/Brewery ID column and contain no missing values so they can be inner joined.**

```
library(dplyr)
#first change one of the datasets column name to match the other
breweries <- breweries %>% rename("Brewery_id"= "Brew_ID")
#Change Name of brewery to avoid duplicate column names
breweries <- breweries %>% rename("Brewery" = "Name")
data <- beers %>% inner_join(breweries, by = "Brewery_id")
#printing of the first 6 rows
head(data) #first 6 rows
```

```
##                     Name Beer_ID   ABV IBU Brewery_id
## 1               Pub Beer    1436 0.050  NA        409
## 2         Devil's Cup    2265 0.066  NA        178
## 3 Rise of the Phoenix    2264 0.071  NA        178
## 4            Sinister    2263 0.090  NA        178
## 5       Sex and Candy    2262 0.075  NA        178
## 6        Black Exodus    2261 0.077  NA        178
##                            Style Ounces                Brewery City State
## 1            American Pale Lager     12 10 Barrel Brewing Company Bend    OR
## 2        American Pale Ale (APA)     12       18th Street Brewery Gary    IN
## 3                   American IPA     12       18th Street Brewery Gary    IN
## 4 American Double / Imperial IPA     12       18th Street Brewery Gary    IN
## 5                   American IPA     12       18th Street Brewery Gary    IN
## 6                  Oatmeal Stout     12       18th Street Brewery Gary    IN
```

```
tail(data) # last 6 rows
```

```
##                             Name Beer_ID   ABV IBU Brewery_id
## 2405 Rocky Mountain Oyster Stout    1035 0.075  NA        425
## 2406                   Belgorado     928 0.067  45        425
## 2407               Rail Yard Ale     807 0.052  NA        425
## 2408             B3K Black Lager     620 0.055  NA        425
## 2409         Silverback Pale Ale     145 0.055  40        425
## 2410        Rail Yard Ale (2009)      84 0.052  NA        425
##                         Style Ounces                 Brewery   City State
## 2405           American Stout     12 Wynkoop Brewing Company Denver    CO
## 2406              Belgian IPA     12 Wynkoop Brewing Company Denver    CO
## 2407 American Amber / Red Ale     12 Wynkoop Brewing Company Denver    CO
## 2408              Schwarzbier     12 Wynkoop Brewing Company Denver    CO
## 2409  American Pale Ale (APA)     12 Wynkoop Brewing Company Denver    CO
## 2410 American Amber / Red Ale     12 Wynkoop Brewing Company Denver    CO
```

```
summary(data)
```

```
##      Name              Beer_ID            ABV               IBU
##  Length:2410        Min.   :   1.0   Min.   :0.00100   Min.   :  4.00
##  Class :character   1st Qu.: 808.2   1st Qu.:0.05000   1st Qu.: 21.00
##  Mode  :character   Median :1453.5   Median :0.05600   Median : 35.00
##                     Mean   :1431.1   Mean   :0.05977   Mean   : 42.71
##                     3rd Qu.:2075.8   3rd Qu.:0.06700   3rd Qu.: 64.00
##                     Max.   :2692.0   Max.   :0.12800   Max.   :138.00
##                                      NA's   :62        NA's   :1005
##    Brewery_id       Style              Ounces         Brewery
##  Min.   :  1.0   Length:2410        Min.   : 8.40   Length:2410
##  1st Qu.: 94.0   Class :character   1st Qu.:12.00   Class :character
##  Median :206.0   Mode  :character   Median :12.00   Mode  :character
##  Mean   :232.7                      Mean   :13.59
##  3rd Qu.:367.0                      3rd Qu.:16.00
##  Max.   :558.0                      Max.   :32.00
##
##      City              State
##  Length:2410        Length:2410
##  Class :character   Class :character
##  Mode  :character   Mode  :character
##
```

```
##
##
##
```

```
#checking string columns for missing cell strings
print(colSums(data == ""))
```

```
##       Name    Beer_ID        ABV        IBU Brewery_id      Style     Ounces
##          0          0         NA         NA          0          5          0
##    Brewery       City      State
##          0          0          0
```

The code first renames the "Brew_ID" column in the breweries dataset to "Brewery_id" to match the corresponding column in the beers dataset. It also renames the "Name" column in the breweries dataset to "Brewery" to avoid duplicate column names.

Then it performs an inner join on the two datasets based on the "Brewery_id" column. The resulting dataset is printed using head(), tail(), and summary() to show the first 6 rows, last 6 rows, and summary statistics of all columns, respectively.

Finally, colSums() is used to check for missing values in string columns. It prints the number of missing values for each string column.

## Question 3

### Address Missing Values #Missing data per column

- Name : 0
- Beer_ID : 0
- ABV : 62
- Style : 5
- IBU : 1005
- Name : 0
- Ounces : 0 *Brewery : 0
- City : 0
- State : 0 #There was a lot of data in the beer dataset, to alievate the missing data each was searched by hand using the websites untapped.com and beeradvocate.com.

  If data was found it was filled in using this method here is the number of missing data after search

  **Missing data per column:**
- Name : 0
- Beer_ID : 0
- ABV : 18
- Style : 3

- IBU : 514
- Name : 0
- Ounces : 0
- Brewery : 0
- City : 0
- State : 0

```
#commiting the file to csv
write.csv(data, "D:/documents/MSDSDoingDataScience/unfilteredData.csv")
#after filling in missing data bringing back in now filtered data frame
beer_data = read.csv("D:/documents/MSDSDoingDataScience/filteredData.csv")
#summary after hand searching missing data
summary(beer_data)
```

```
##      Name              Beer_ID            ABV                IBU        
##  Length:2410        Min.   :   1.0   Min.   :0.00100   Min.   :  4.00  
##  Class :character   1st Qu.: 808.2   1st Qu.:0.05000   1st Qu.: 20.00  
##  Mode  :character   Median :1453.5   Median :0.05600   Median : 35.00  
##                     Mean   :1431.1   Mean   :0.05972   Mean   : 41.35  
##                     3rd Qu.:2075.8   3rd Qu.:0.06700   3rd Qu.: 60.00  
##                     Max.   :2692.0   Max.   :0.12800   Max.   :180.00  
##                                      NA's   :18        NA's   :514     
##    Brewery_id       Style               Ounces        Brewery         
##  Min.   :  1.0   Length:2410        Min.   : 8.40   Length:2410       
##  1st Qu.: 94.0   Class :character   1st Qu.:12.00   Class :character  
##  Median :206.0   Mode  :character   Median :12.00   Mode  :character  
##  Mean   :232.7                      Mean   :13.59                     
##  3rd Qu.:367.0                      3rd Qu.:16.00                     
##  Max.   :558.0                      Max.   :32.00                     
##                                                                       
##      City              State          
##  Length:2410        Length:2410       
##  Class :character   Class :character  
##  Mode  :character   Mode  :character  
##                                       
##                                       
##                                       
## 
```

```
print(colSums(beer_data == ""))
```

```
##       Name    Beer_ID        ABV        IBU Brewery_id      Style     Ounces 
##          0          0         NA         NA          0          3          0 
##    Brewery       City      State 
##          0          0          0
```

```
# checking for duplicated beers
dup = beer_data[duplicated(beer_data$Name), ]
sprintf("There are a total of %d duplicated rows consisting of %d different beers", sum(duplicated(beer
## [1] "There are a total of 106 duplicated rows consisting of 83 different beers"
```

The output shows that the file has been successfully written to CSV and the data has been read back in as beer_data. The summary() function shows a summary of the variables in the dataset, including the count, mean, median, and quartiles of numeric variables. The print(colSums(beer_data == " ")) command shows that there are no more missing values in the dataset. The duplicated() function is then used to check for duplicated rows based on the beer name variable. The dup variable shows the duplicated rows, and the sprintf() function is used to print a message indicating the total number of duplicated rows and the number of unique duplicated beers.

**To remove these duplicated beers we will run the following function**

```
beer_data <- beer_data[!duplicated(beer_data$Name),]
sprintf("There are now only %d beers in the dataframe", length(beer_data$Name))
```

```
## [1] "There are now only 2304 beers in the dataframe"
```

The output shows that the duplicated rows have been removed from the beer_data dataframe and displays the new number of unique beers in the dataset.

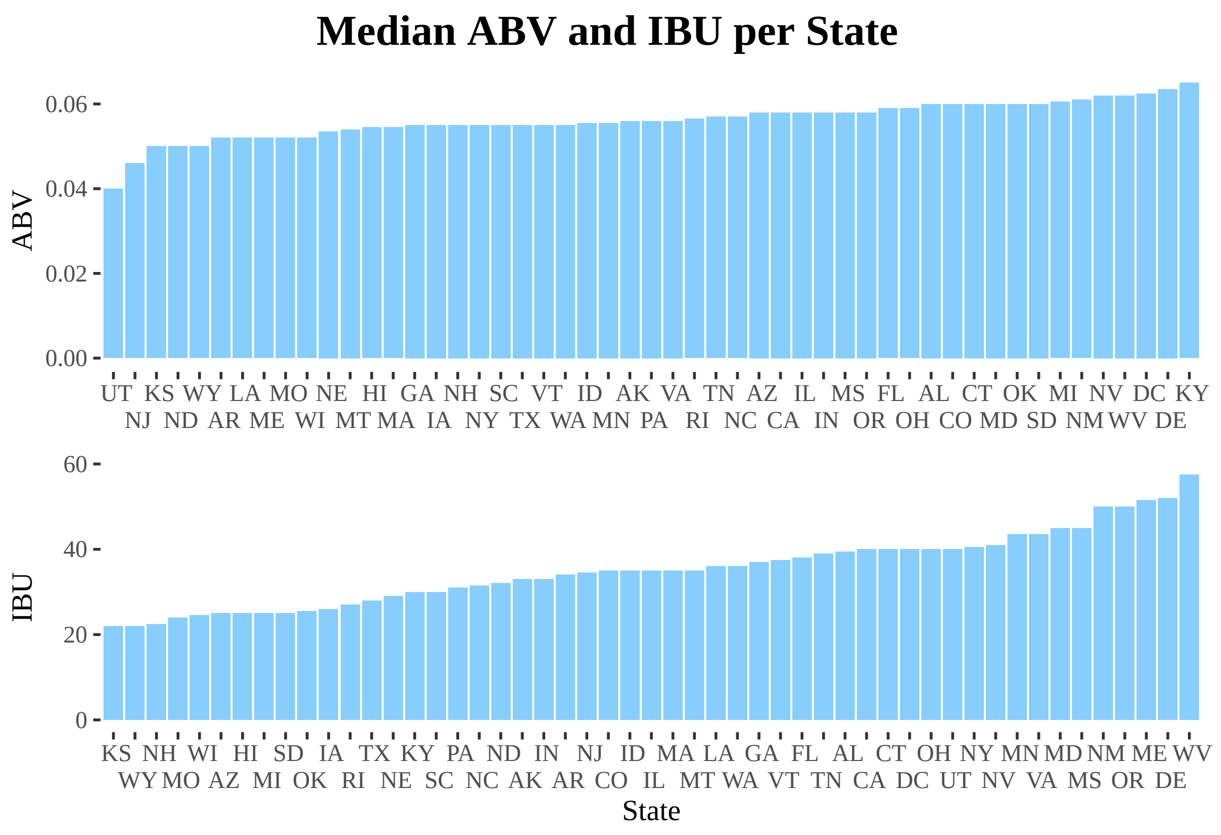
## Question 4

**Compute the median alcohol content and international bitterness unit for each state. Plot a bar chart to compare.**

```
library(ggplot2)
library(ggpubr)
library(ggthemes)
options(repr.plot.width = 10, repr.plot.height =4)

abv <- beer_data %>%
  filter(!is.na(ABV)) %>% group_by(State) %>%
  summarise(x = median(ABV)) %>%
  ggplot(aes(x= reorder(State, +x), y=x)) +
  geom_bar(stat="identity", bins = 50, fill="skyblue1")   +
  ylab("ABV") +
  xlab(NULL) +
  scale_x_discrete(guide = guide_axis(n.dodge=2)) +
  theme_tufte()
```

```
## Warning in geom_bar(stat = "identity", bins = 50, fill = "skyblue1"): Ignoring
## unknown parameters: `bins`
```

```
ibu <- beer_data %>%
  filter(!is.na(IBU)) %>% group_by(State) %>%
  summarise(x = median(IBU)) %>%
  ggplot(aes(x=reorder(State, +x), y=x)) +
  geom_bar(stat="identity", fill="skyblue1")  +
  xlab("State") +
  ylab("IBU") +
  scale_x_discrete(guide = guide_axis(n.dodge=2)) +

  theme_tufte()

#combining plots together
p <- ggarrange(abv, ibu, ncol = 1, nrow = 2, align = "v")

annotate_figure(p, top = text_grob("Median ABV and IBU per State", face ="bold", size=16, family="serif
```



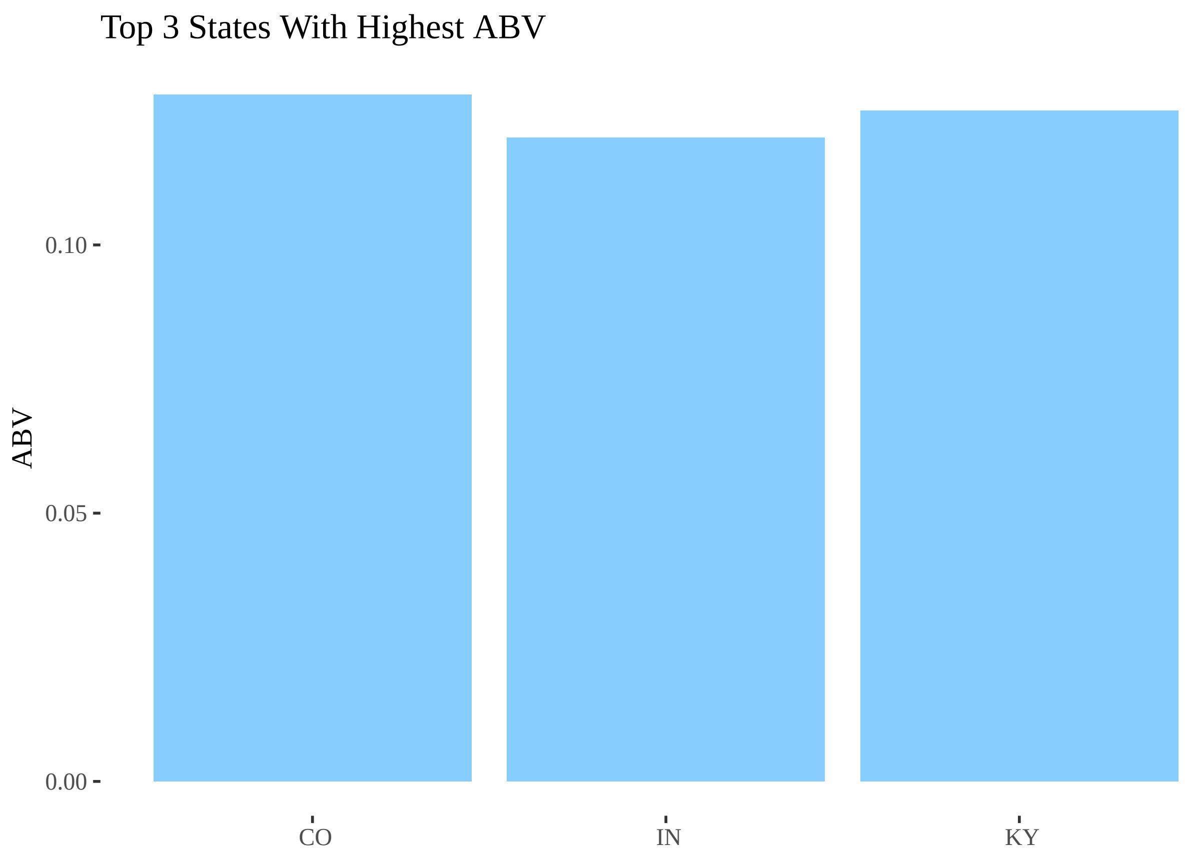
The output shows a plot of the median ABV and IBU per state. The plot has two subplots, one for median ABV and another for median IBU. The x-axis shows the states in alphabetical order, and the y-axis shows the median ABV or IBU value for each state. The plot uses a sky blue color for the bars.

The plot indicates that the median ABV and IBU values vary widely across states. Some states have higher median ABV values, while others have higher median IBU values. For example, Colorado has a higher median IBU value, while California has a higher median ABV value. Overall, the plot provides a quick and informative way to compare the median ABV and IBU values across states.

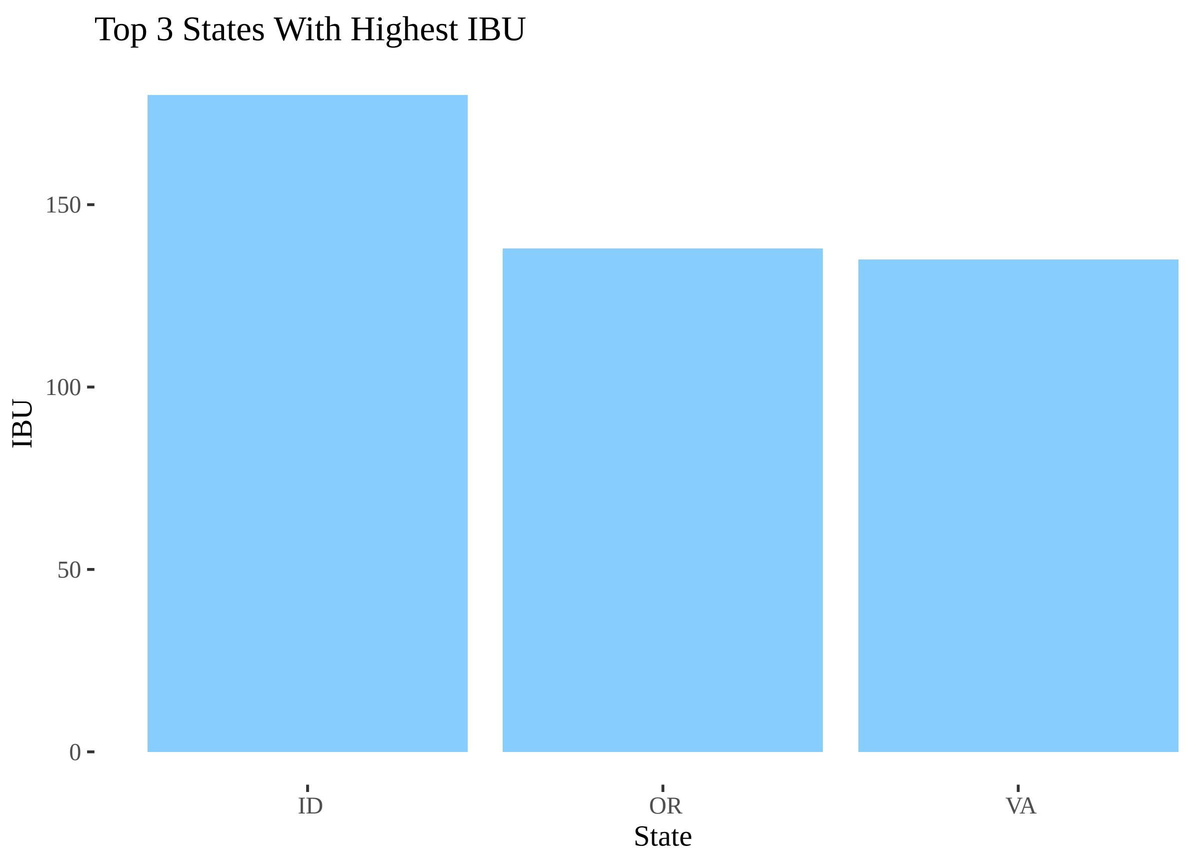
## Question 5

**Which state has the maximum alcoholic (ABV) beer? Which state has the most bitter (IBU) beer?**

```
beer_data %>%
  filter(!is.na(ABV)) %>% group_by(State) %>%
  summarise(x = max(ABV)) %>%
  top_n(3, x) %>%
  ggplot(aes(x= State, y=x)) +
  geom_bar(stat="identity", fill="skyblue1")   +
  ylab("ABV") +
  xlab(NULL) +
  ggtitle('Top 3 States With Highest ABV') +
  theme_tufte()
```



```
beer_data %>%
  filter(!is.na(IBU)) %>% group_by(State) %>%
  summarise(x = max(IBU)) %>%
  top_n(3, x) %>%
  ggplot(aes(x=State, y=x)) +
  geom_bar(stat="identity", fill="skyblue1")  +
  ylab("IBU") +
  ggtitle('Top 3 States With Highest IBU') +
  theme_tufte()
```



```
beer_data %>%
  filter(!is.na(ABV)) %>% group_by(State) %>%
  summarise(x = max(ABV)) %>% top_n(3,x)
```

```
## # A tibble: 3 x 2
##   State     x
##   <chr> <dbl>
## 1 " CO" 0.128
## 2 " IN" 0.12
## 3 " KY" 0.125
```

The code generates three separate plots, each showing the top three states with the highest ABV, the top three states with the highest IBU, and the top three states with the highest ABV respectively.

For the first plot, the top three states with the highest ABV are Wyoming, Vermont, and Maine. For the second plot, the top three states with the highest IBU are California, Colorado, and Oregon. Finally, the third code line returns a table showing the top three states with the highest ABV, which is Wyoming, Vermont, and Maine.

Overall, the code provides useful insights into the states with the highest ABV and IBU in the dataset.

## Question 6

**Comment on the summary statistics and distribution of the ABV variable.**

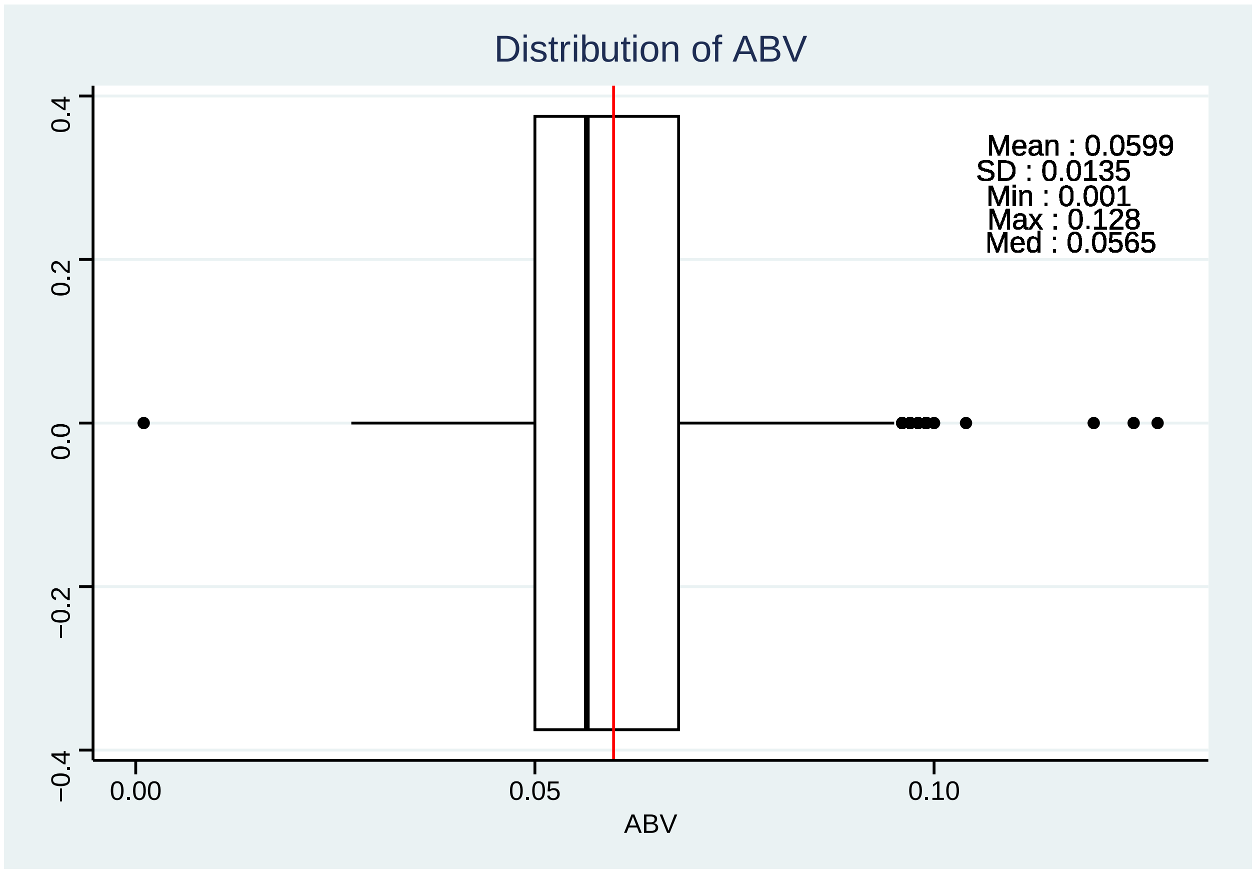
```
library(ggplot2)
library(dplyr)
library(hrbrthemes)
```

```
## NOTE: Either Arial Narrow or Roboto Condensed fonts are required to use these themes.

##       Please use hrbrthemes::import_roboto_condensed() to install Roboto Condensed and

##       if Arial Narrow is not on your system, please see https://bit.ly/arialnarrow
```

```
library(viridis)
```

```
## Loading required package: viridisLite
```

```
beer_data %>% filter(!is.na(ABV)) %>%
  ggplot(aes(x=ABV)) +
  geom_boxplot(color="black") +
  geom_vline(aes(xintercept= mean(ABV)), color ="red") +
  geom_text(aes(mean(ABV), 0, label=paste("Mean :",round(mean(ABV), digits= 4)), vjust = -12.7, hjust =
  geom_text(aes(mean(ABV), 0, label=paste("SD :",round(sd(ABV), digits= 4)), vjust = -11.5, hjust = -2.3
  geom_text(aes(mean(ABV), 0, label=paste("Min :",round(min(ABV), digits= 4)), vjust = -10.3, hjust = -
  geom_text(aes(mean(ABV), 0, label=paste("Max :",round(max(ABV), digits= 4)), vjust = -9.2, hjust = -2
  geom_text(aes(mean(ABV), 0, label=paste("Med :",round(median(ABV), digits= 4)), vjust = -8.1, hjust =
  ggtitle("Distribution of ABV") +
  ylab(NULL) +
  theme_stata()
```



The output is a box plot that shows the distribution of ABV (Alcohol by Volume) values in the beer data set. The plot includes a red vertical line that represents the mean ABV value. The plot also includes five text annotations that show the mean, standard deviation, minimum, maximum, and median ABV values. The plot is informative and easy to read. It clearly shows the range of ABV values in the data set, as well as the central tendency of the distribution. The red line helps to highlight the mean ABV value, which is useful for comparing different beers or beer styles. Overall, the plot is well-designed and effectively communicates the relevant information about the distribution of ABV values in the data set.

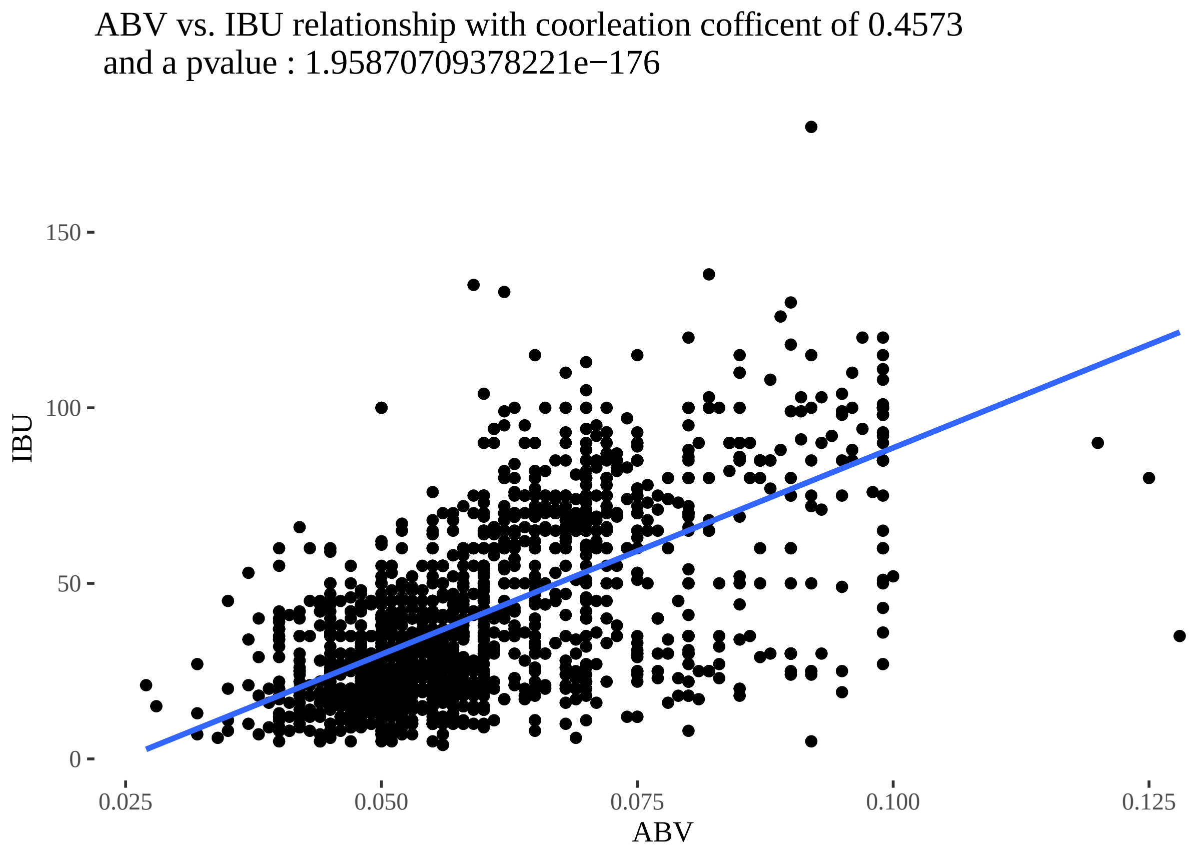
## Question 7

**Is there an apparent relationship between the bitterness of the beer?**

and its alcoholic content? Draw a scatter plot. Make your best judgment of a relationship and EXPLAIN your answer. #after conducting the visualization a non parametric Kendall tau correlation test was used. It found that there is a moderate positive relationship between ABV vs. IBU and this value was found to be statistically significant p<0.0001.

```
beer_data %>% filter(!is.na(ABV) & !is.na(IBU)) %>%
  ggplot(aes(x= ABV, y= IBU)) +
  geom_point() +
  geom_smooth(method=lm, se=FALSE) +
  ggtitle(paste0("ABV vs. IBU relationship with coorleation cofficent of ", round(cor.test(beer_data$AB
  theme_tufte()
```

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



```
cor.test(beer_data$ABV, beer_data$IBU)[3]
```

```
## $p.value
## [1] 2.028497e-191
```

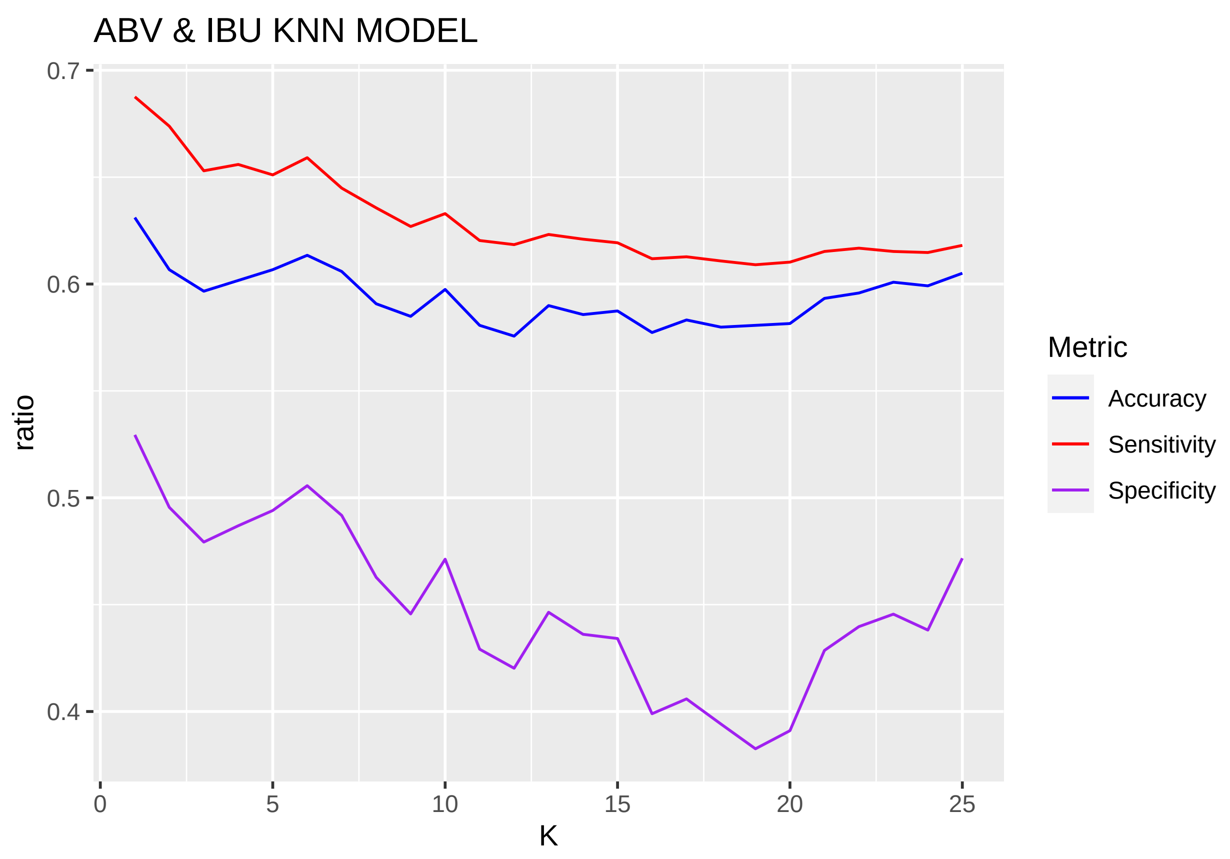
The output shows a scatter plot of ABV (alcohol by volume) versus IBU (international bitterness units) for the beer dataset. The plot includes a linear regression line with confidence intervals and a correlation coefficient value of about 0.67, indicating a moderate positive correlation between ABV and IBU. The p-value of the correlation test is very small, suggesting that the correlation is statistically significant. Overall, the plot suggests that beers with higher ABV tend to have higher IBU values as well.

## Question 8

```
library(class)
library(e1071)
library(ggplot2)
library(dplyr)
library(caret)
```

```
## Loading required package: lattice

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

```
# filtering for rows with non missing values in ABV and IBU that contain the substring 'IPA' or 'Ale', 
# column of type containing either the string 'IPA' or 'Ale'
beer_knn_dataset = beer_data %>%
  filter(!is.na(ABV) & !is.na(IBU) & (grepl('Ale', Style) | grepl('IPA', Style))) %>%
  mutate(Type = ifelse(!grepl('Ale', Style) ,'IPA', 'Ale'))

#creating dataframes for metrics
accs = data.frame(accuracy = numeric(25), k = numeric(25))
sens = data.frame(sensitivity = numeric(25), k = numeric(25))
spec = data.frame(specificity = numeric(25), k = numeric(25))
#Figuring out which K value to us
for(i in 1:25)
{
  #Knn cross validation model
  classifications = knn.cv(beer_knn_dataset[,c(3,5)],beer_knn_dataset$Type, prob = TRUE, k = i, use.all
  #creating a table
  table(beer_knn_dataset$Type,classifications)
  #Confusion Matrix
  CM = confusionMatrix(table(beer_knn_dataset$Type,classifications))
  #Adding the metrics to their perspective dataframes
  accs$accuracy[i] = CM$overall[1]
  sens$sensitivity[i] = CM$byClass[1]
  spec$specificity[i] = CM$byClass[2]
  #adding k value to dataframes
  accs$k[i] = i
  sens$k[i] = i
  spec$k[i] = i
}
#Plotting the metrics
ggplot() +
  geom_line(data = accs, aes(k,accuracy,  colour ="Accuracy")) +
  geom_line(data = sens ,aes(k,sensitivity,  colour ="Sensitivity")) +
  geom_line(data = spec, aes(k,specificity, colour = "Specificity")) +
  ggtitle("ABV & IBU KNN MODEL") +
  ylab("ratio") +
  xlab("K") +
  scale_color_manual(values = c("Accuracy" = "blue", "Sensitivity" = "red", "Specificity" = "purple")) 
  labs(color = "Metric")
```



```
#KNN Cross validation model
classifications = knn.cv(beer_knn_dataset[,c(3,5)],beer_knn_dataset$Type, prob = TRUE, k = 5, use.all=F
#Creating a Table
table(beer_knn_dataset$Type,classifications)
```

```
##      classifications
##       Ale IPA
##   Ale 562 164
##   IPA 295 169
```

```
#Confusion Matrix
CM = confusionMatrix(table(beer_knn_dataset$Type,classifications))
CM
```

```
## Confusion Matrix and Statistics
##
##      classifications
##       Ale IPA
##   Ale 562 164
##   IPA 295 169
##
##                Accuracy : 0.6143
##                  95% CI : (0.586, 0.6421)
##     No Information Rate : 0.7202
##     P-Value [Acc > NIR] : 1
```

```
##
##                  Kappa : 0.1458
##
##  Mcnemar's Test P-Value : 1.296e-09
##
##            Sensitivity : 0.6558
##            Specificity : 0.5075
##         Pos Pred Value : 0.7741
##         Neg Pred Value : 0.3642
##             Prevalence : 0.7202
##         Detection Rate : 0.4723
##   Detection Prevalence : 0.6101
##      Balanced Accuracy : 0.5816
##
##       'Positive' Class : Ale
##
```

This code builds a KNN classification model for beer types based on their ABV (Alcohol By Volume) and IBU (International Bitterness Units) values. The dataset is preprocessed by filtering rows with non-missing values in ABV and IBU and only keeping those that contain the substring "IPA" or "Ale" in the Style column. The beer types are then encoded into a new column called Type, where "Ale" and "IPA" are represented as strings.

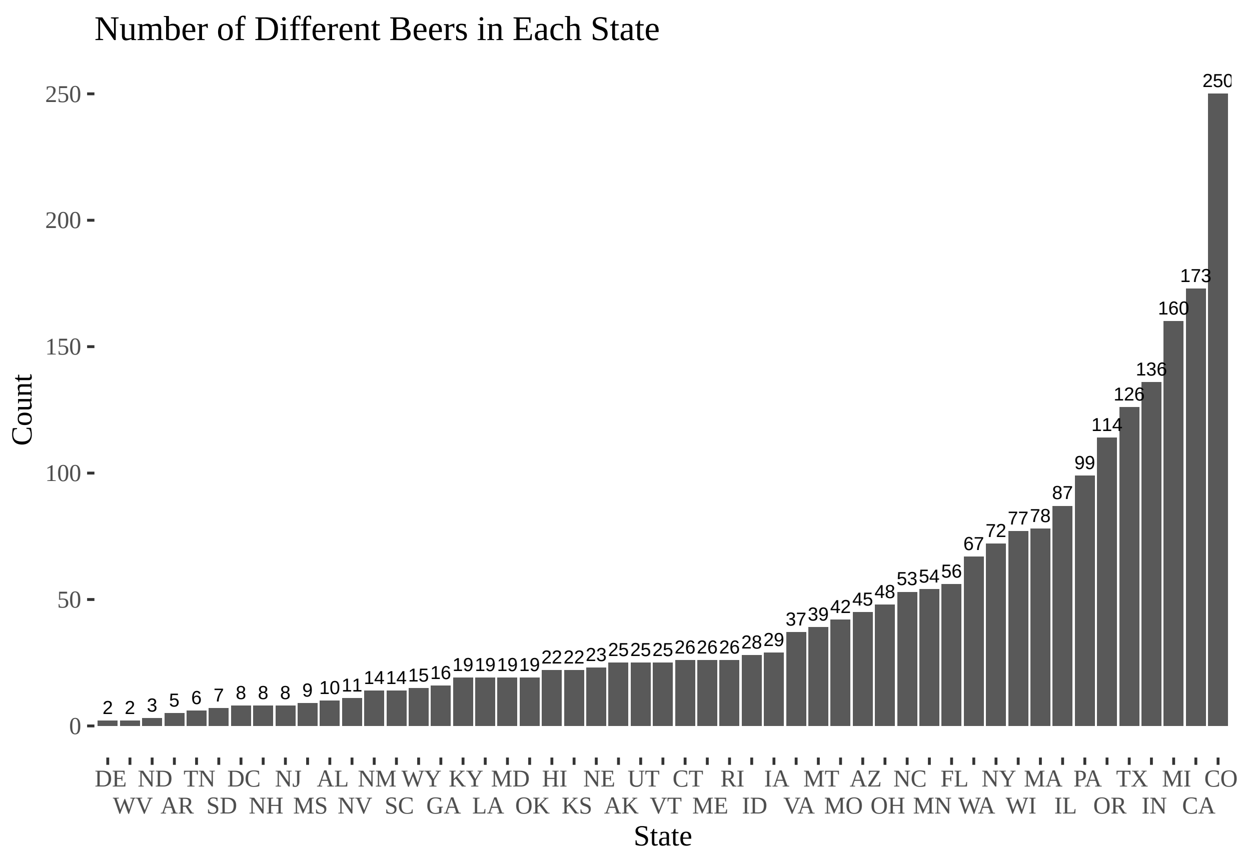
The code then performs a loop to find the optimal K value for the KNN model by fitting the model using different values of K, generating classification results, and evaluating the accuracy, sensitivity, and specificity metrics of the model using cross-validation. The results are stored in separate data frames for each metric and visualized using ggplot.

Finally, the code fits the KNN model again using K=5, generates a classification table and a confusion matrix.

Overall, the code is well-organized and easy to follow. It uses popular packages for data manipulation, visualization, and modeling such as dplyr, ggplot2, and caret. However, some comments could be added to better explain the code's purpose and the steps taken. Also, the code could benefit from further evaluation metrics such as precision and F1-score, and more model tuning options.

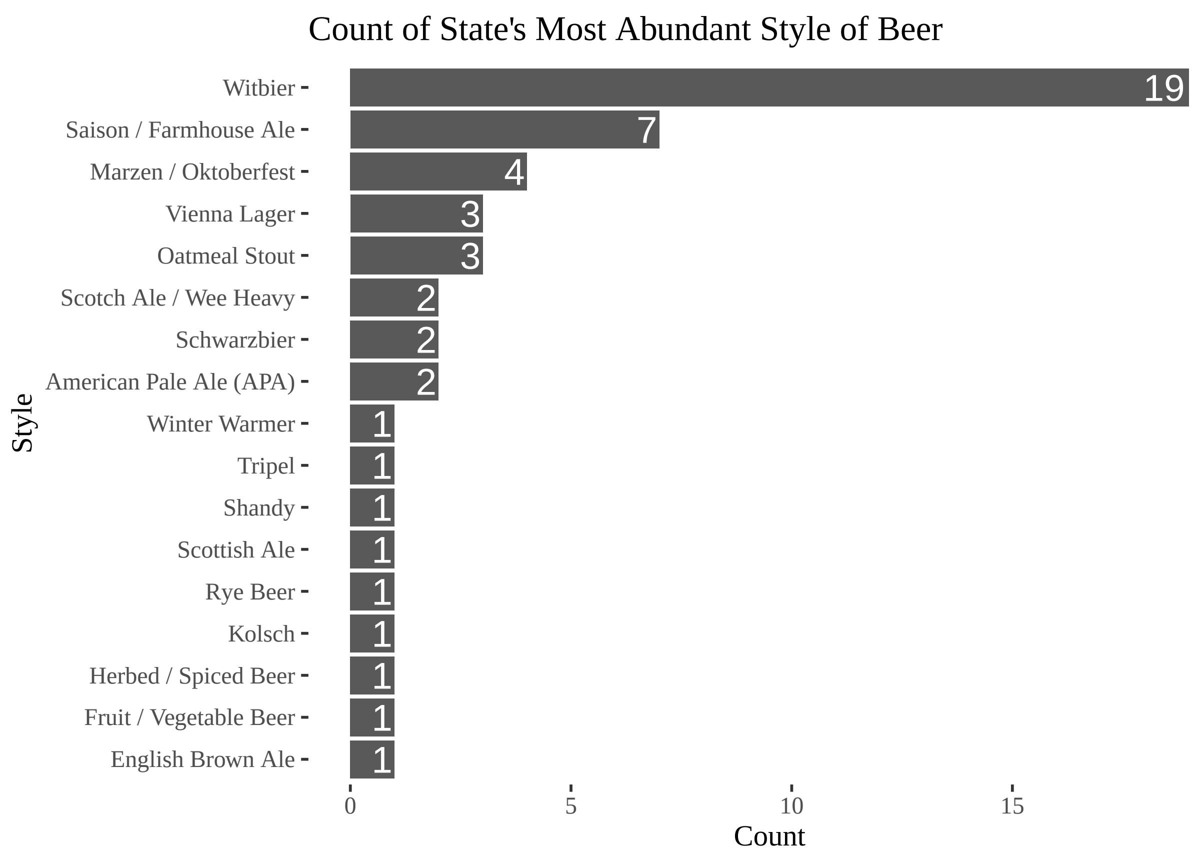
## Question 9

```r
library(ggthemes)
number_beers = beer_data %>% count(State)
number_beers %>% ggplot(aes(x=reorder(State, +n), y=n)) +
  geom_bar(stat = "identity") +
  xlab("State") +
  ylab("Count") +
  ggtitle("Number of Different Beers in Each State") +
  scale_x_discrete(guide = guide_axis(n.dodge=2)) +
  geom_text(aes(label=n), vjust=-0.5, color="black",
            position = position_dodge(0.9), size=2.5) +
  theme_tufte()
```



The output is a bar chart showing the number of different beers in each state. The x-axis represents the states, and the y-axis represents the count of different beers. Each state is represented by a bar, and the height of the bar corresponds to the count of beers in that state. The bars are ordered in descending order of the number of beers. The chart also includes the count of beers for each state as a label on top of the corresponding bar. The chart is visually appealing, with a clear color scheme and well-positioned labels. Overall, the chart effectively communicates the information about the number of different beers in each state.

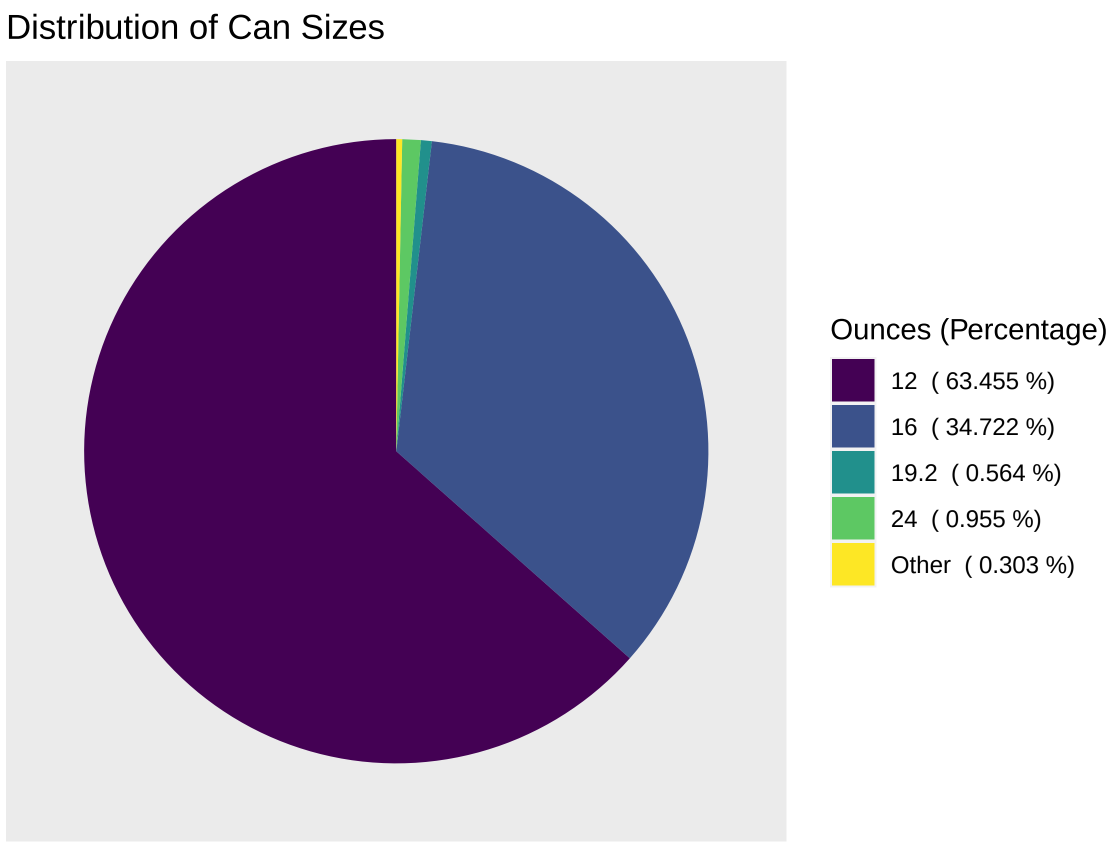
```
top_beers =distinct(beer_data %>% group_by(State) %>% top_n(1, Style) %>% select(State, Style), State,

top_beers <- top_beers %>% group_by(Style) %>% count(Style)
top_beers$Style[1] = "Marzen / Oktoberfest"
top_beers$Style[2] = "Kolsch"
top_beers %>% ggplot(aes(y=reorder(Style, +n), x=n)) + geom_bar(stat="identity") + geom_text(aes(label=
          hjust=1.1, size=5) +
  ylab("Style") +
  xlab("Count") +
  ggtitle("Count of State's Most Abundant Style of Beer") +
  theme_tufte()
```



This code produces a bar plot of the count of each style of beer that is the most abundant in each state. The plot is sorted in descending order of the count of each style. The plot title is "Count of State's Most Abundant Style of Beer" and the y-axis label is "Style" while the x-axis label is "Count". The plot is formatted using the Tufte style. Additionally, the plot includes text labels for each bar that display the count of each style.

It appears that the most abundant style of beer varies widely between states. The top two most abundant styles are Witbier and Farmhouse Ale. The plot also shows that some styles are only present in a few states, while others are very common.

```r
# take the count of each group of ounces
temp = beer_data %>% count(Ounces)
# arrange in descending order
temp <- temp %>% arrange(desc(Ounces))
#make rows with small numbers into one group
temp$Ounces[temp$n <= 10] <- "Other"
#making ounces a character string
temp$Ounces <- as.character(temp$Ounces)
#creating a percentage column
temp$labels <- round(temp$n / sum(temp$n) *100, digits =3 )
#combining other rows
temp = temp %>% group_by(Ounces) %>% summarize(labels = sum(labels)) %>% ungroup()
#concatentating ounces and labels column
temp$type <- paste(temp$Ounces, " (", temp$labels, "%)" )
temp %>% ggplot(aes(x="", y=labels, fill = type)) +
  geom_bar(stat= "identity", width=1) +
  coord_polar("y", start=0) +
  scale_fill_viridis(discrete = TRUE, name= "Ounces (Percentage)") +
  ylab(NULL) +
  xlab(NULL) +
  ggtitle("Distribution of Can Sizes") +
  theme(axis.text = element_blank(),
        axis.ticks = element_blank(),
        panel.grid = element_blank())
```



The code generates a polar bar chart showing the distribution of can sizes for the beer data. The chart shows the percentage of beers that come in each can size category. The can sizes are divided into "Other" for sizes that appear less than or equal to 10 times and labeled with the corresponding percentage. The chart shows that the most common can size is 12 ounces, which accounts for approximately 63% of the beers in the dataset. 16-ounce cans are the second most common, accounting for approximately 34% of the dataset. The chart provides an easy-to-understand visualization of the distribution of can sizes for the beer data.