Intro to R

#### Execute the following cells to load the libraries

library(ggplot2)  
library(dplyr)

##   
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':  
##   
## filter, lag

## The following objects are masked from 'package:base':  
##   
## intersect, setdiff, setequal, union

#### Load the food texture dataset

file = 'Codes/Data/food-texture.csv'  
#file = 'Data/food-texture.csv'  
foodData = read.csv(file, header = TRUE, row.names = 1, stringsAsFactors = FALSE)  
head(foodData, 2)

## Oil Density Crispy Fracture Hardness  
## B110 16.5 2955 10 23 97  
## B136 17.7 2660 14 9 139

str(foodData)

## 'data.frame': 50 obs. of 5 variables:  
## $ Oil : num 16.5 17.7 16.2 16.7 16.3 19.1 18.4 17.5 15.7 16.4 ...  
## $ Density : int 2955 2660 2870 2920 2975 2790 2750 2770 2955 2945 ...  
## $ Crispy : int 10 14 12 10 11 13 13 10 11 11 ...  
## $ Fracture: int 23 9 17 31 26 16 17 26 23 24 ...  
## $ Hardness: int 97 139 143 95 143 189 114 63 123 132 ...

#### Modify Crispy column to reflect high (0) or low (1) crispiness

foodData = foodData %>% mutate(Crispylevel = ifelse(Crispy > 11, 'High', 'Low'))

##### Change Crispy and Crispylevel columns to factor (categorical) type

# Continuous features -> Oil, Density, Hardness, Fracture  
# Categorical features -> Crispy (8 levels 8 through 15) and Crispylevel (2 levels 0 and 1)  
# Crispy is a categorical feature with an order  
str(foodData)

## 'data.frame': 50 obs. of 6 variables:  
## $ Oil : num 16.5 17.7 16.2 16.7 16.3 19.1 18.4 17.5 15.7 16.4 ...  
## $ Density : int 2955 2660 2870 2920 2975 2790 2750 2770 2955 2945 ...  
## $ Crispy : int 10 14 12 10 11 13 13 10 11 11 ...  
## $ Fracture : int 23 9 17 31 26 16 17 26 23 24 ...  
## $ Hardness : int 97 139 143 95 143 189 114 63 123 132 ...  
## $ Crispylevel: chr "Low" "High" "High" "Low" ...

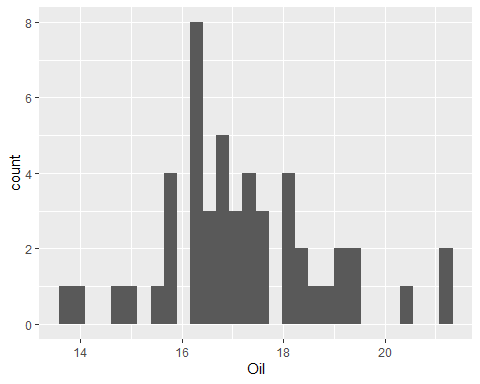
categorical\_cols = c('Crispy', 'Crispylevel')  
foodData[categorical\_cols] = lapply(foodData[categorical\_cols], as.factor)  
str(foodData)

## 'data.frame': 50 obs. of 6 variables:  
## $ Oil : num 16.5 17.7 16.2 16.7 16.3 19.1 18.4 17.5 15.7 16.4 ...  
## $ Density : int 2955 2660 2870 2920 2975 2790 2750 2770 2955 2945 ...  
## $ Crispy : Factor w/ 9 levels "7","8","9","10",..: 4 8 6 4 5 7 7 4 5 5 ...  
## $ Fracture : int 23 9 17 31 26 16 17 26 23 24 ...  
## $ Hardness : int 97 139 143 95 143 189 114 63 123 132 ...  
## $ Crispylevel: Factor w/ 2 levels "High","Low": 2 1 1 2 2 1 1 2 2 2 ...

#### Visualize the OilPercentage feature using a histogram

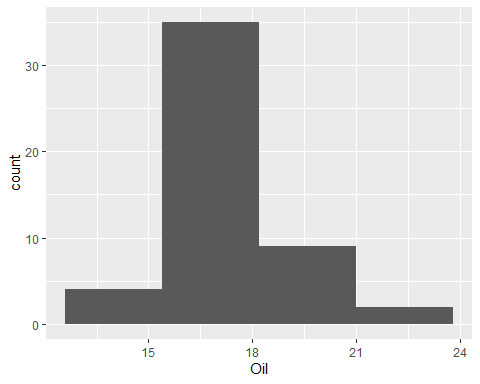
p=ggplot(data=foodData)+  
 geom\_histogram(aes(x=Oil))  
p

## `stat\_bin()` using `bins = 30`. Pick better value with `binwidth`.



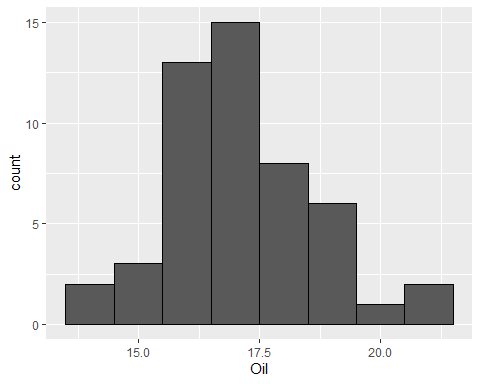
##binwidth

p=ggplot(data=foodData)+  
 geom\_histogram(aes(x=Oil),binwidth=2.8)  
p

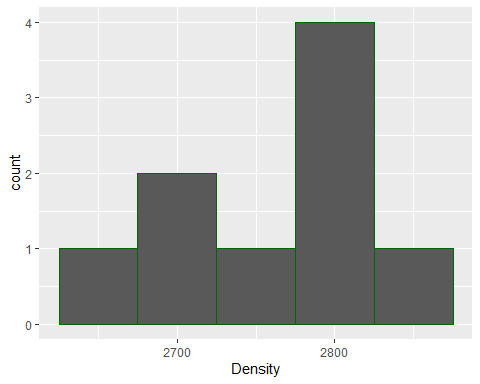


#color

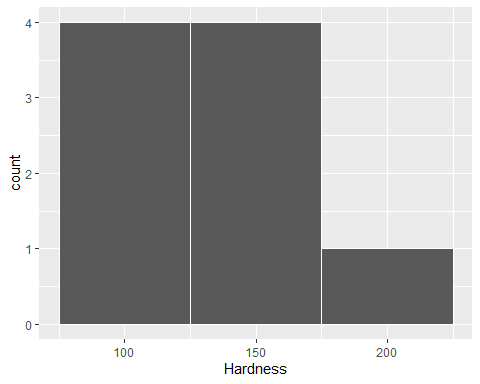
p=ggplot(data=foodData)+  
 geom\_histogram(aes(x=Oil),binwidth=1,color='black')  
p



p=ggplot(data=foodData[foodData$Crispy==13,])+  
 geom\_histogram(aes(x=Density),binwidth = 50,color='darkgreen')  
p

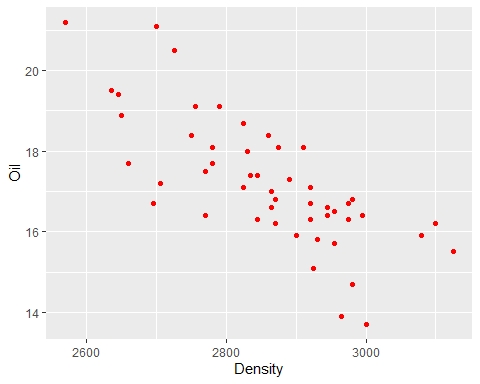


p=ggplot(data=foodData[foodData$Crispy==13,])+  
 geom\_histogram(aes(x=Hardness),binwidth = 50,color='white')  
p

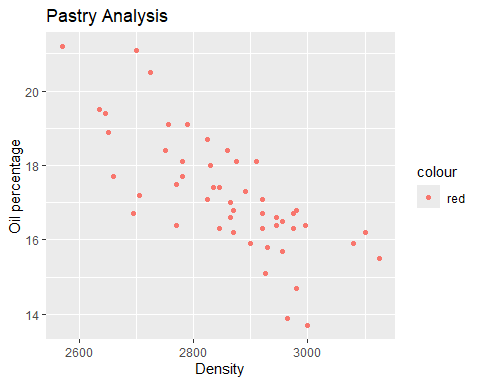


#### Scatter plot between OilPercentage and Density

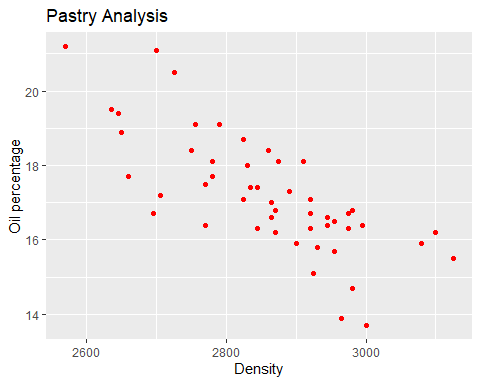
p=ggplot(data=foodData)+  
 geom\_point(aes(x=Density,y=Oil),color='red')  
p



p=ggplot(data=foodData)+  
 geom\_point(aes(x=Density,y=Oil,color='red'))+  
 labs(x='Density',y='Oil percentage',title='Pastry Analysis')  
p

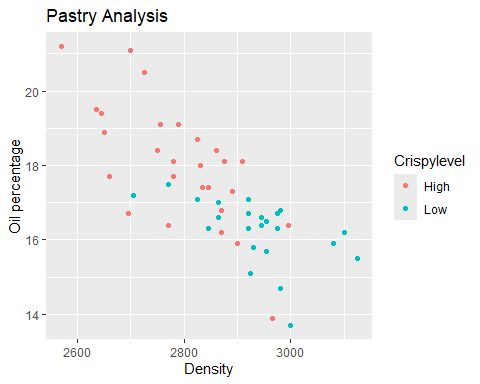


p=ggplot(data=foodData)+  
 geom\_point(aes(x=Density,y=Oil),color='red')+  
 labs(x='Density',y='Oil percentage',title='Pastry Analysis')  
p



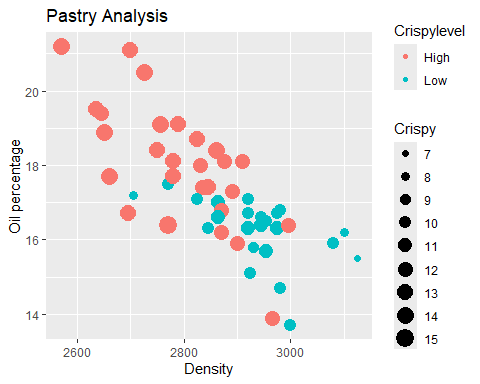
#### Scatter plot between OilPercentage and Density color coded with Crispylevel

p=ggplot(data=foodData)+  
 geom\_point(aes(x=Density,y=Oil,color=Crispylevel))+  
 labs(x='Density',y='Oil percentage',title='Pastry Analysis')  
p



p=ggplot(data=foodData)+  
 geom\_point(aes(x=Density,y=Oil,color=Crispylevel,size=Crispy))+  
 labs(x='Density',y='Oil percentage',title='Pastry Analysis')  
p

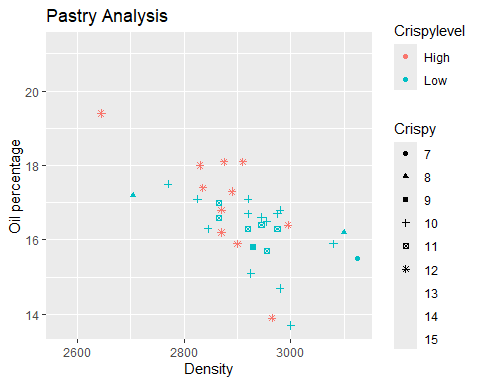
## Warning: Using size for a discrete variable is not advised.



p=ggplot(data=foodData)+  
 geom\_point(aes(x=Density,y=Oil,color=Crispylevel,shape=Crispy))+  
 labs(x='Density',y='Oil percentage',title='Pastry Analysis')  
p

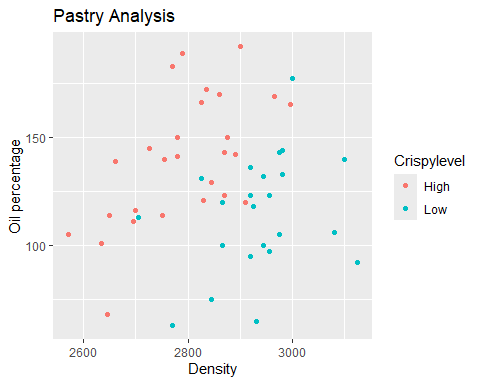
## Warning: The shape palette can deal with a maximum of 6 discrete values because more  
## than 6 becomes difficult to discriminate  
## ℹ you have requested 9 values. Consider specifying shapes manually if you need  
## that many have them.

## Warning: Removed 16 rows containing missing values or values outside the scale range  
## (`geom\_point()`).



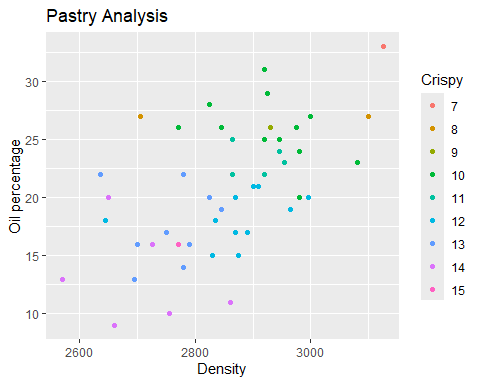
#### Scatter plot between Density and Hardness

p=ggplot(data=foodData)+  
 geom\_point(aes(x=Density,y=Hardness,color=Crispylevel))+  
 labs(x='Density',y='Oil percentage',title='Pastry Analysis')  
p



#### Scatter plot between Density and Fracture

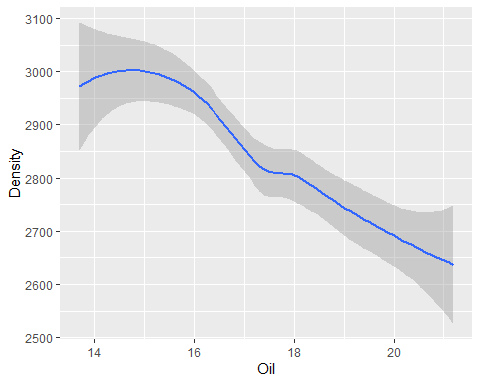
p=ggplot(data=foodData)+  
 geom\_point(aes(x=Density,y=Fracture,color=Crispy))+  
 labs(x='Density',y='Oil percentage',title='Pastry Analysis')  
p



#### Smooth line plot using ggplot

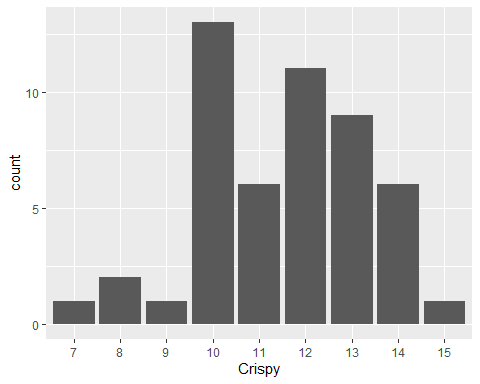
p=ggplot(data=foodData)+  
 geom\_smooth(aes(x=Oil,y=Density))  
p

## `geom\_smooth()` using method = 'loess' and formula = 'y ~ x'



#### Barplot for Crispy

p=ggplot(data=foodData)+  
 geom\_bar(aes(x=Crispy))  
p

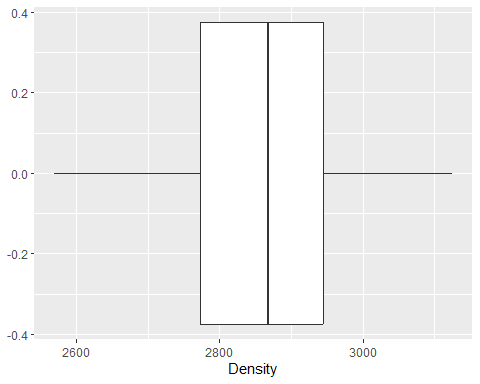


#### In-built functions for dataframes

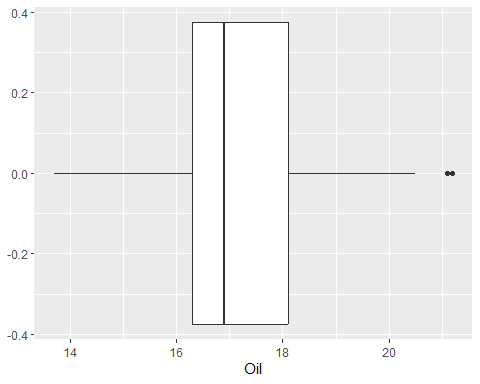
# Mean oil percentage across all samples  
  
# Mean-centering of OilPercentage  
  
# Sum of the squared deviation from the mean  
  
# Average of the squared deviation from the mean  
  
# Variance of OilPercentage  
  
# Standard deviation of OilPercentage

#### Box plot using ggplot with color coding

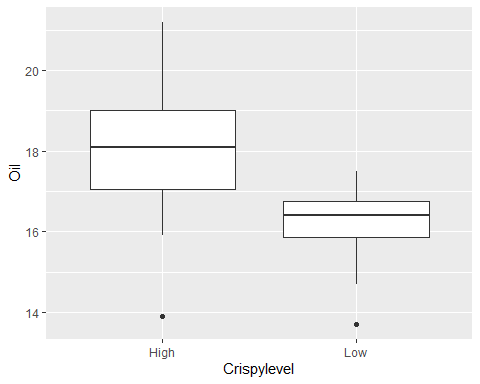
p=ggplot(data=foodData)+  
 geom\_boxplot(aes(x=Density))  
p



p=ggplot(data=foodData)+  
 geom\_boxplot(aes(x=Oil))  
p



p=ggplot(data=foodData)+  
 geom\_boxplot(aes(x=Crispylevel,y=Oil))  
p



p=ggplot(data=foodData[foodData$Crispylevel=='High',])+  
 geom\_histogram(aes(x=Density),binwidth = 100,color='blue')  
p

