PROG8430-Assignment 4

Mohammed Saifullah

2023-04-13

Loading necessary packages

```r
#Load packages
if(!require(pastecs)){install.packages("pastecs")}
```

```
## Loading required package: pastecs
```

```r
library("pastecs")

if(!require(lattice)){install.packages("lattice")}
```

```
## Loading required package: lattice
```

```
## Warning: package 'lattice' was built under R version 4.2.3
```

```r
library("lattice")

if(!require(tinytex)){install.packages("tinytex")}
```

```
## Loading required package: tinytex
```

```r
library("tinytex")

if(!require(corrgram)){install.packages("corrgram")}
```

```
## Loading required package: corrgram
```

```
##
## Attaching package: 'corrgram'
```

```
## The following object is masked from 'package:lattice':
##
##     panel.fill
```

```r
library("corrgram")
```

Part 1 - Preliminary and Exploratory

##1.1 Appending initials to all column names

```
getwd() #verify working directory
```

```
## [1] "E:/Big Data Solution Architecture/PROG8430 - Data Analysis Mathematics, Algorithms and Modeling
```

```
#Read the text data file into a Data Frame
MailOrder_MS <- read.table("PROG8430_Assign04_23W.txt", sep=',', header = TRUE)
#concatenating initial 'MS' to all column names
colnames(MailOrder_MS) <- paste(colnames(MailOrder_MS), "MS", sep = "_")
#Display first 5 rows of the dataset just to verify loading and name transformation is  successful
head(MailOrder_MS, 5)
```

```
##   DL_MS VN_MS PG_MS CS_MS ML_MS DM_MS HZ_MS    CR_MS WT_MS
## 1   8.1   324     5    13   313     C     N  Sup Del   216
## 2   8.4   135     2    13   830     I     N  Sup Del   160
## 3   8.6   391     3    12   304     C     N  Sup Del    25
## 4  11.3   245     6     7  1258     C     N  Sup Del    67
## 5   5.4   321     1     2   221     C     N Def Post    14
```

```
#Transform String as Factor variable
MailOrder_MS <- as.data.frame(unclass(MailOrder_MS), stringsAsFactors = TRUE)
#Checking Data Structure
str(MailOrder_MS)
```

```
## 'data.frame':    487 obs. of  9 variables:
##  $ DL_MS: num  8.1 8.4 8.6 11.3 5.4 9.4 8.2 9.4 9.3 9.7 ...
##  $ VN_MS: int  324 135 391 245 321 397 390 252 355 159 ...
##  $ PG_MS: int  5 2 3 6 1 2 6 2 4 1 ...
##  $ CS_MS: int  13 13 12 7 2 8 13 8 2 12 ...
##  $ ML_MS: int  313 830 304 1258 221 1002 655 1367 675 888 ...
##  $ DM_MS: Factor w/ 2 levels "C","I": 1 2 1 1 1 2 1 2 1 1 ...
##  $ HZ_MS: Factor w/ 2 levels "H","N": 2 2 2 2 2 2 2 2 2 2 ...
##  $ CR_MS: Factor w/ 2 levels "Def Post","Sup Del": 2 2 2 2 1 2 2 2 2 2 ...
##  $ WT_MS: num  216 160 25 67 14 47 7 6 30 177 ...
```

##1.2 Examining the Data

Checking data Summary and stat

```
summary(MailOrder_MS)
```

```
##      DL_MS            VN_MS           PG_MS            CS_MS
##  Min.   : 1.800   Min.   : 85.0   Min.   :-2.000   Min.   : 0.000
##  1st Qu.: 7.400   1st Qu.:263.0   1st Qu.: 2.000   1st Qu.: 5.000
##  Median : 8.500   Median :322.0   Median : 3.000   Median : 8.000
##  Mean   : 8.464   Mean   :318.6   Mean   : 2.951   Mean   : 9.228
##  3rd Qu.: 9.550   3rd Qu.:371.0   3rd Qu.: 4.000   3rd Qu.:13.000
##  Max.   :14.400   Max.   :495.0   Max.   : 9.000   Max.   :24.000
##      ML_MS        DM_MS   HZ_MS        CR_MS         WT_MS      
##  Min.   :  35.0   C:344   H: 69   Def Post:201   Min.   :  0.1  
##  1st Qu.: 444.5   I:143   N:418   Sup Del :286   1st Qu.: 33.0  
##  Median : 697.0                                  Median : 87.0  
##  Mean   : 754.0                                  Mean   :107.1  
##  3rd Qu.:1021.5                                  3rd Qu.:157.5  
##  Max.   :1967.0                                  Max.   :500.0  
```

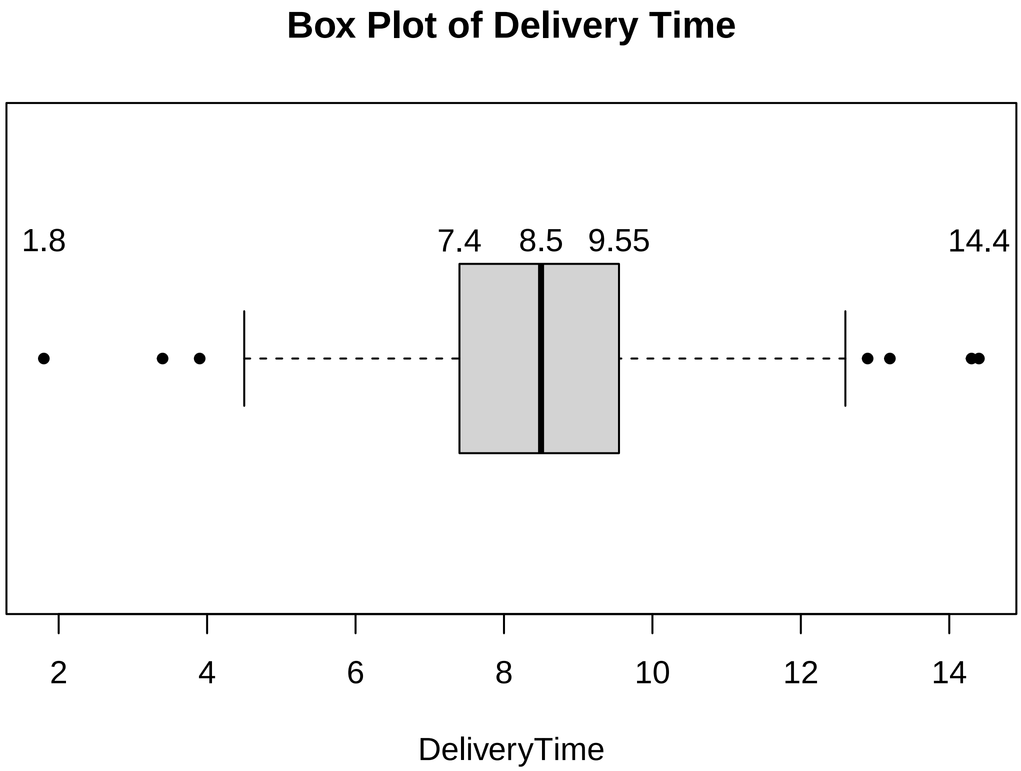
```
stat.desc(MailOrder_MS)
```

```
##                      DL_MS          VN_MS        PG_MS        CS_MS
## nbr.val       487.00000000    487.0000000  487.0000000  487.0000000
## nbr.null        0.00000000      0.0000000    0.0000000    2.0000000
## nbr.na          0.00000000      0.0000000    0.0000000    0.0000000
## min             1.80000000     85.0000000   -2.0000000    0.0000000
## max            14.40000000    495.0000000    9.0000000   24.0000000
## range          12.60000000    410.0000000   11.0000000   24.0000000
## sum          4122.10000000 155143.0000000 1437.0000000 4494.0000000
## median          8.50000000    322.0000000    3.0000000    8.0000000
## mean            8.46427105    318.5687885    2.9507187    9.2279261
## SE.mean         0.07850066      3.3189638    0.0693047    0.2339453
## CI.mean.0.95    0.15424259      6.5212898    0.1361738    0.4596691
## var             3.00106649   5364.5585300    2.3391301   26.6537041
## std.dev         1.73235865     73.2431466    1.5294215    5.1627225
## coef.var        0.20466720      0.2299131    0.5183217    0.5594673
##                       ML_MS DM_MS HZ_MS CR_MS        WT_MS
## nbr.val         487.0000000    NA    NA    NA   487.000000
## nbr.null          0.0000000    NA    NA    NA     0.000000
## nbr.na            0.0000000    NA    NA    NA     0.000000
## min              35.0000000    NA    NA    NA     0.100000
## max            1967.0000000    NA    NA    NA   500.000000
## range          1932.0000000    NA    NA    NA   499.900000
## sum          367194.0000000    NA    NA    NA 52176.100000
## median          697.0000000    NA    NA    NA    87.000000
## mean            753.9917864    NA    NA    NA   107.137782
## SE.mean          18.5981864    NA    NA    NA     4.194176
## CI.mean.0.95     36.5427801    NA    NA    NA     8.240958
## var          168449.6665991    NA    NA    NA  8566.873179
## std.dev         410.4262012    NA    NA    NA    92.557405
## coef.var          0.5443378    NA    NA    NA     0.863910
```

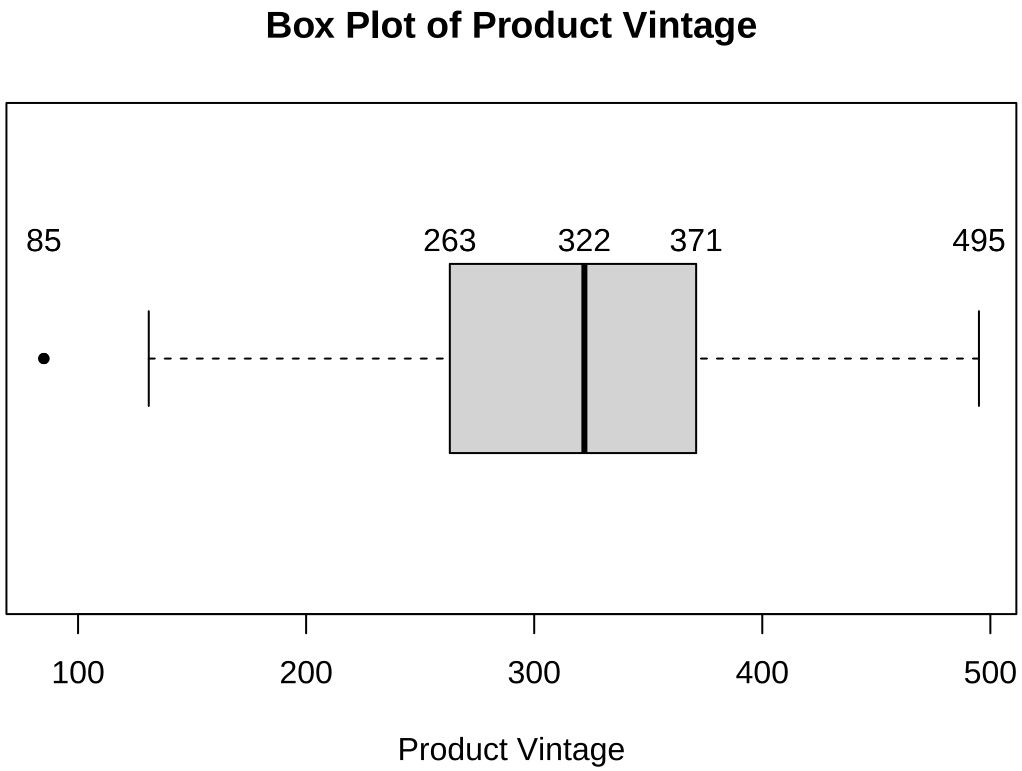
Interpretation: Minimum Package Ordered(PG_MS) is -2, which can not be correct.

Checking Outlier with boxplots and desnisy plots

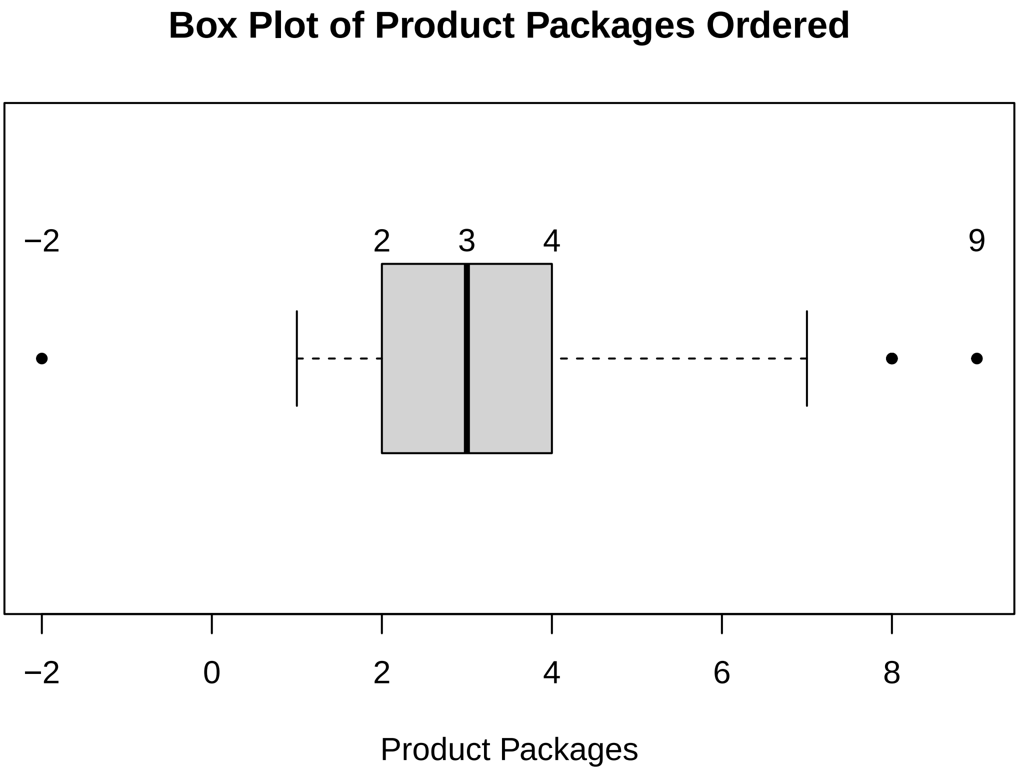
```
# Box Plots to check outliers
boxplot(MailOrder_MS$DL_MS, horizontal = TRUE, pch = 20,
        main="Box Plot of Delivery Time",
        xlab="DeliveryTime")
#display values on the boxplot
text(x=fivenum(MailOrder_MS$DL_MS), labels=fivenum(MailOrder_MS$DL_MS), y=1.25)
```



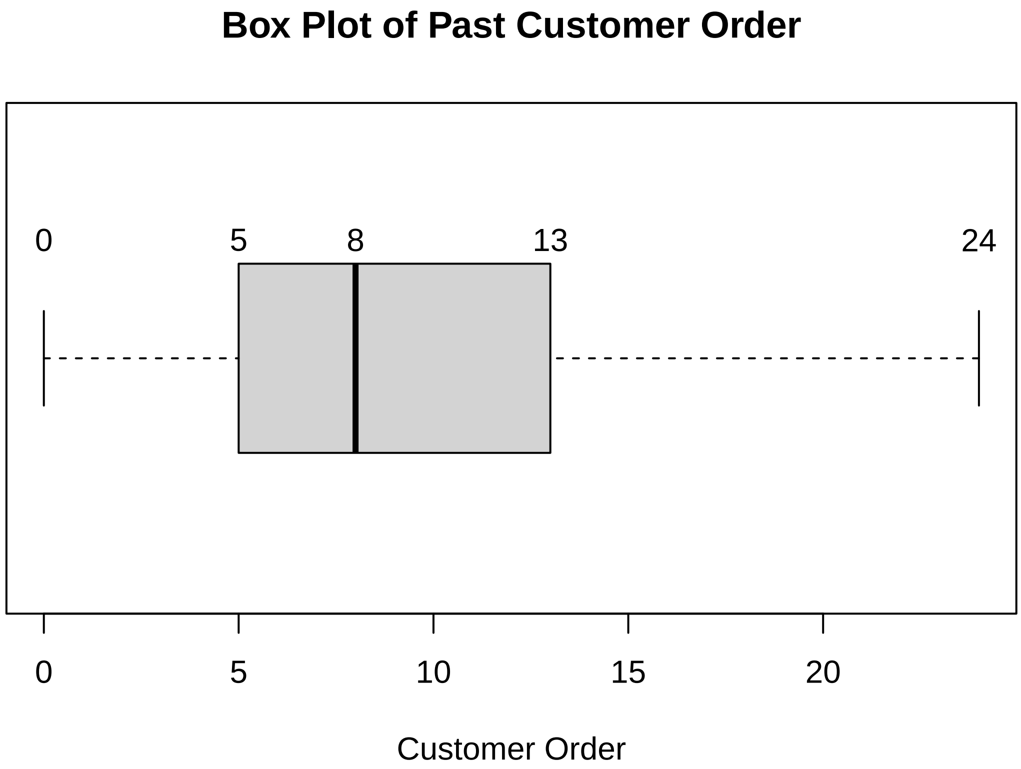
```
boxplot(MailOrder_MS$VN_MS, horizontal = TRUE, pch = 20,
        main="Box Plot of Product Vintage",
        xlab="Product Vintage")
#display values on the boxplot
text(x=fivenum(MailOrder_MS$VN_MS), labels=fivenum(MailOrder_MS$VN_MS), y=1.25)
```



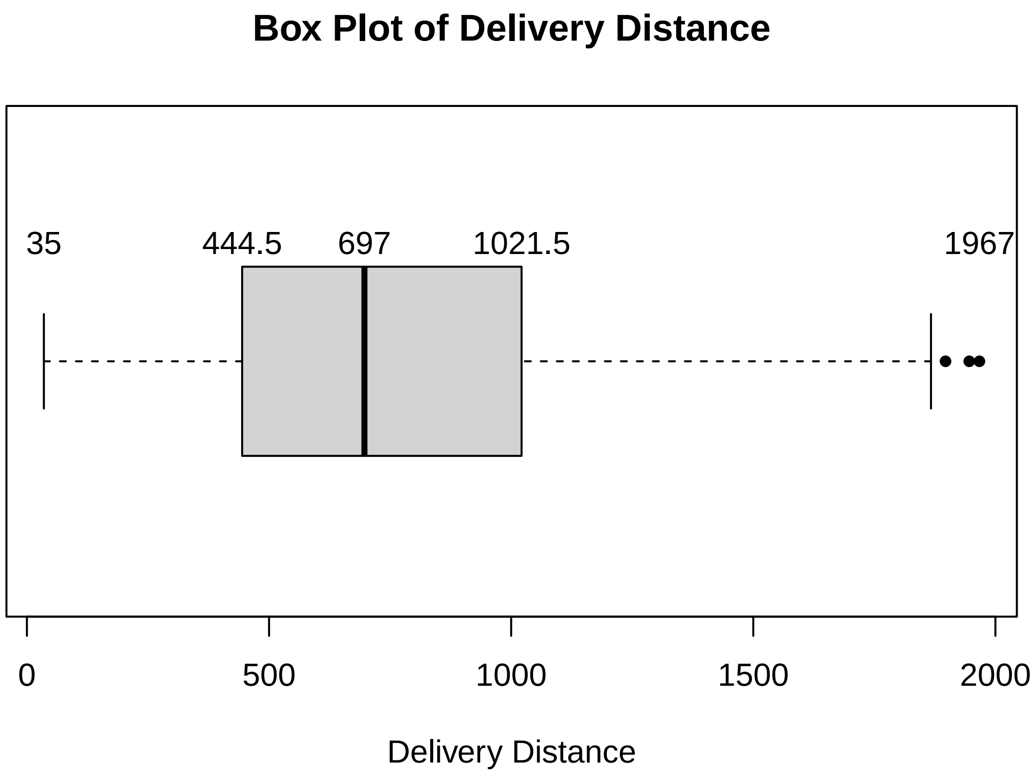
```
#Outliar in Packages Ordered, -2
boxplot(MailOrder_MS$PG_MS, horizontal = TRUE, pch = 20,
        main="Box Plot of Product Packages Ordered",
        xlab="Product Packages")
#display values on the boxplot
text(x=fivenum(MailOrder_MS$PG_MS), labels=fivenum(MailOrder_MS$PG_MS), y=1.25)
```



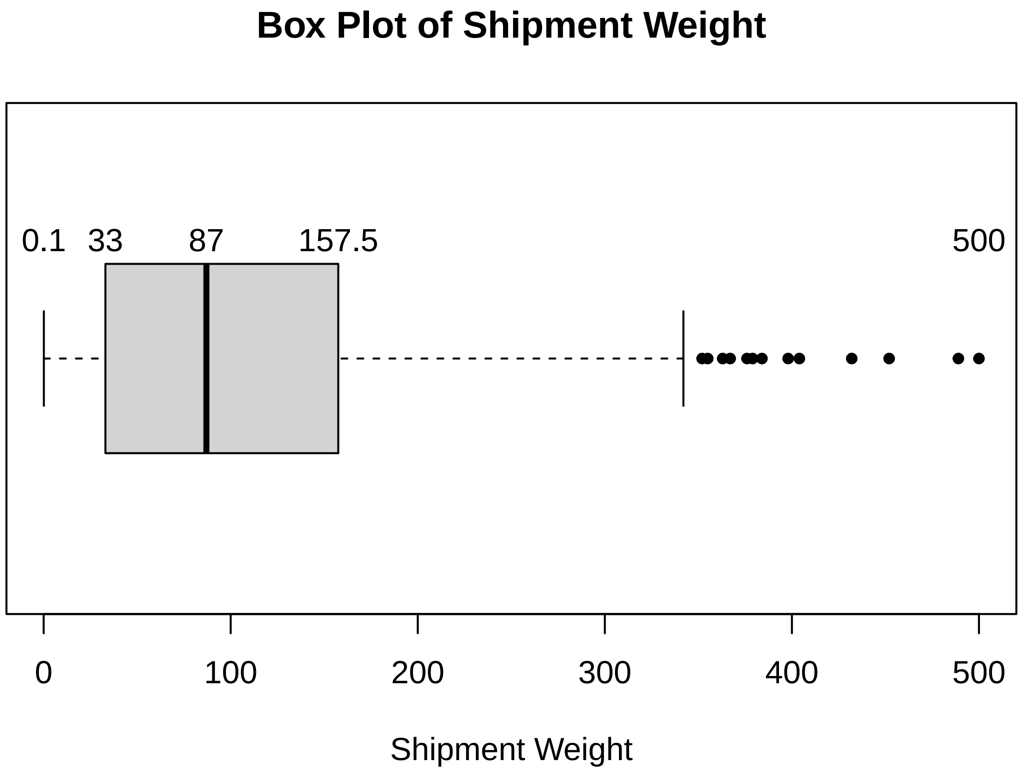
```
boxplot(MailOrder_MS$CS_MS, horizontal = TRUE, pch = 20,
        main="Box Plot of Past Customer Order",
        xlab="Customer Order")
#display values on the boxplot
text(x=fivenum(MailOrder_MS$CS_MS), labels=fivenum(MailOrder_MS$CS_MS), y=1.25)
```



```
boxplot(MailOrder_MS$ML_MS, horizontal = TRUE, pch = 20,
        main="Box Plot of Delivery Distance",
        xlab="Delivery Distance")
#display values on the boxplot
text(x=fivenum(MailOrder_MS$ML_MS), labels=fivenum(MailOrder_MS$ML_MS), y=1.25)
```



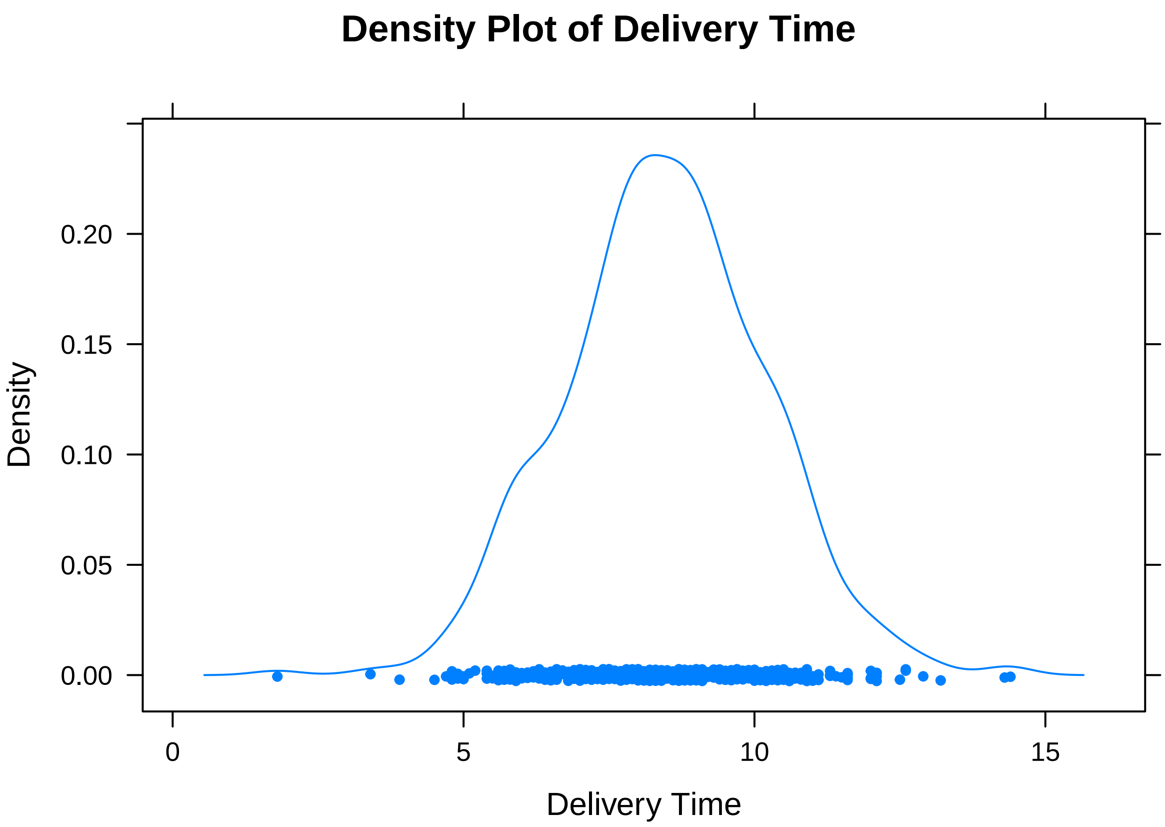
```
boxplot(MailOrder_MS$WT_MS, horizontal = TRUE, pch = 20,
        main="Box Plot of Shipment Weight",
        xlab="Shipment Weight")
#display values on the boxplot
text(x=fivenum(MailOrder_MS$WT_MS), labels=fivenum(MailOrder_MS$WT_MS), y=1.25)
```



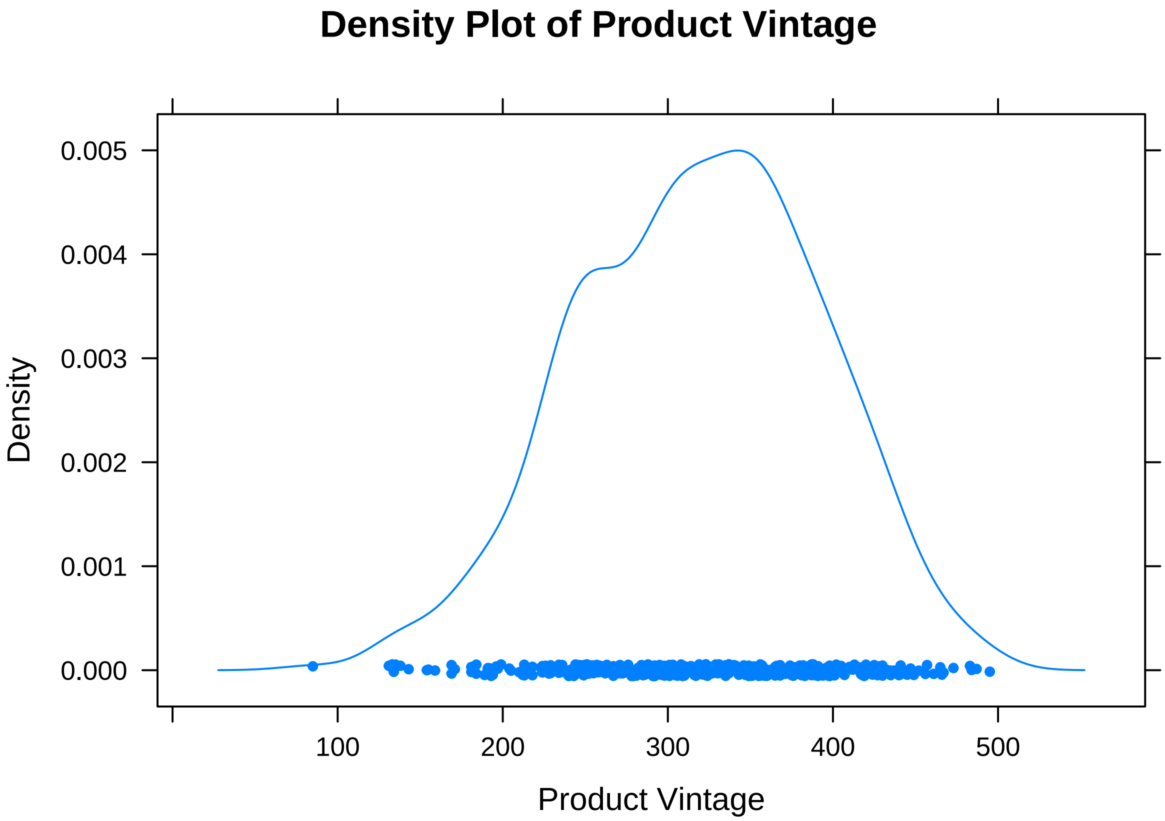
Interpretation:
Clearly Package Ordered has an outlier at -2. Packages ordered should not be negative.

Checking Density plots

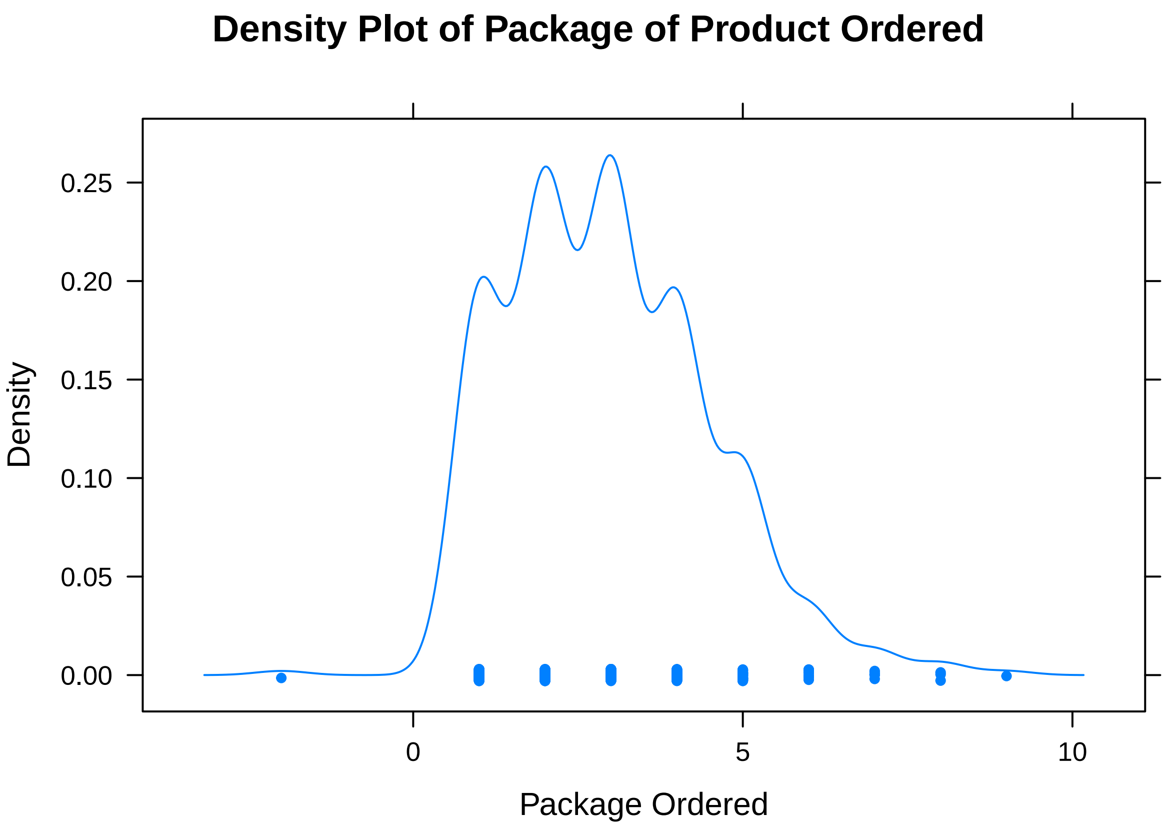
```
densityplot(~ MailOrder_MS$DL_MS, pch = 20,
            main="Density Plot of Delivery Time",
            xlab="Delivery Time")
```



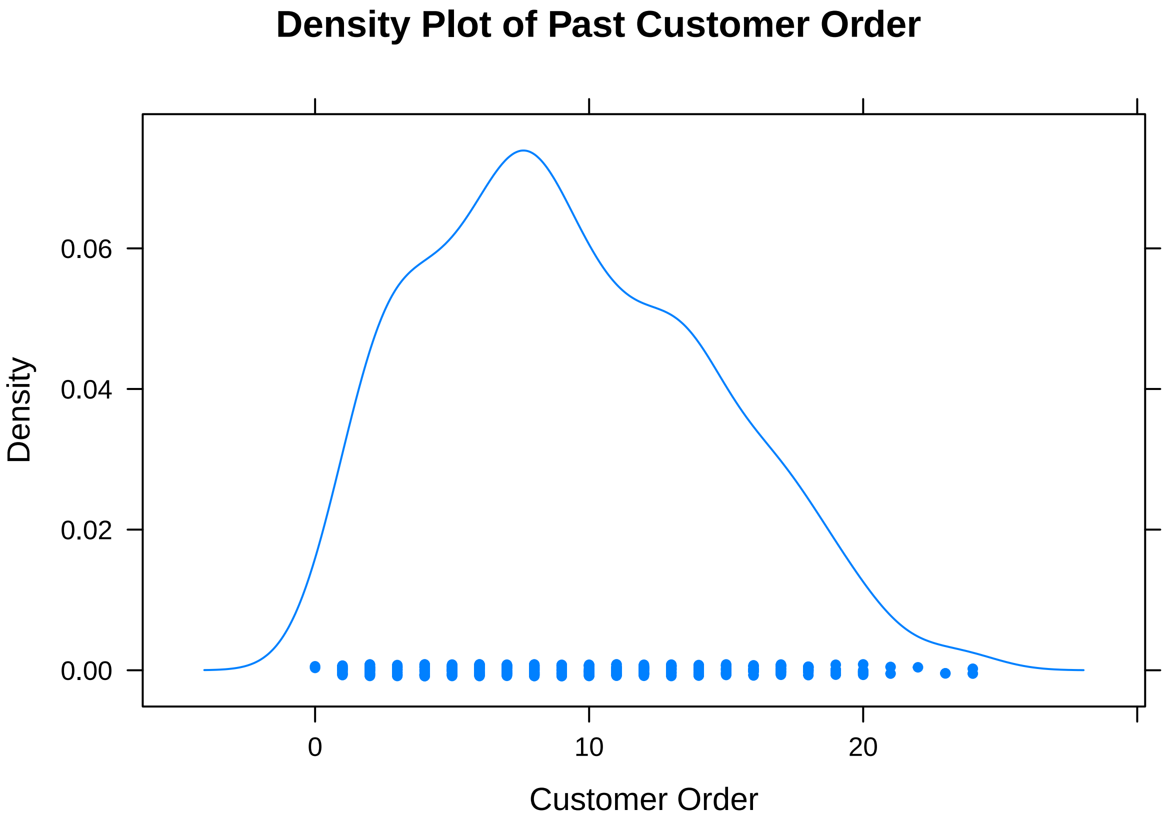
```
densityplot(~ MailOrder_MS$VN_MS, pch = 20,
            main="Density Plot of Product Vintage",
            xlab="Product Vintage")
```



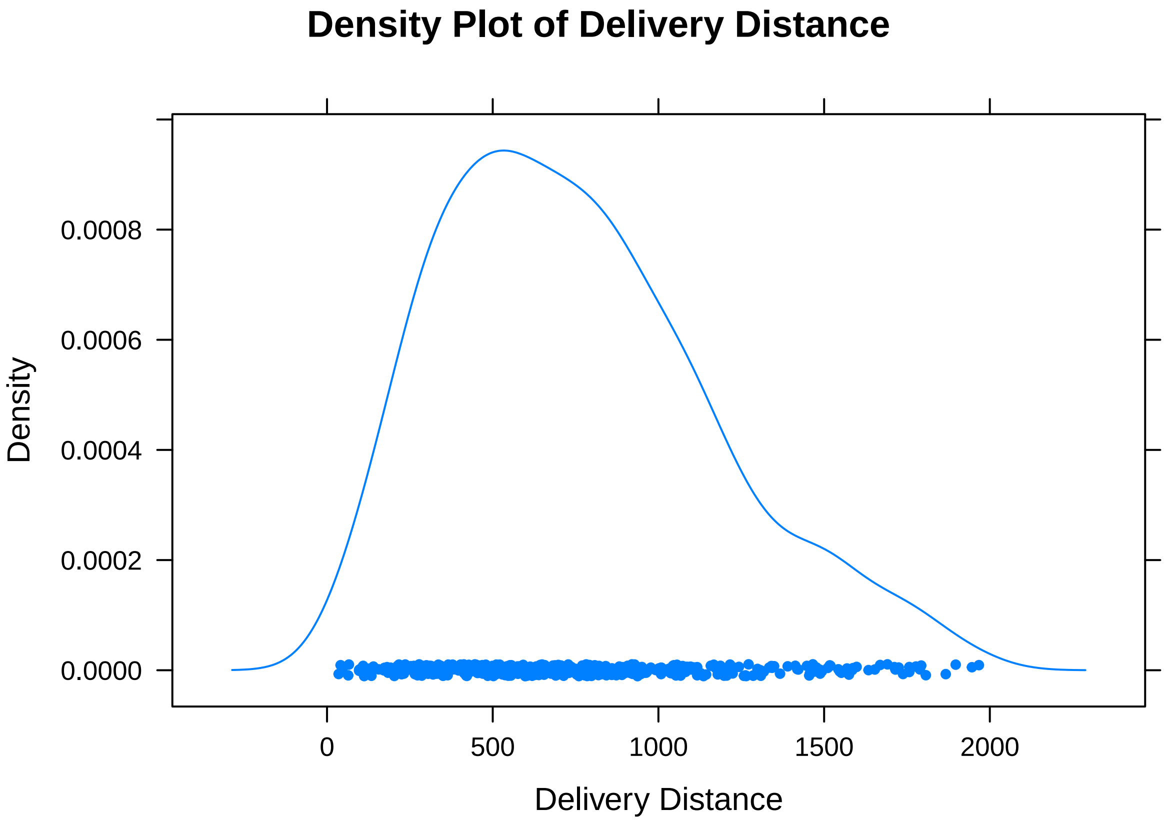
```
densityplot(~ MailOrder_MS$PG_MS, pch = 20,
            main="Density Plot of Package of Product Ordered",
            xlab="Package Ordered")
```



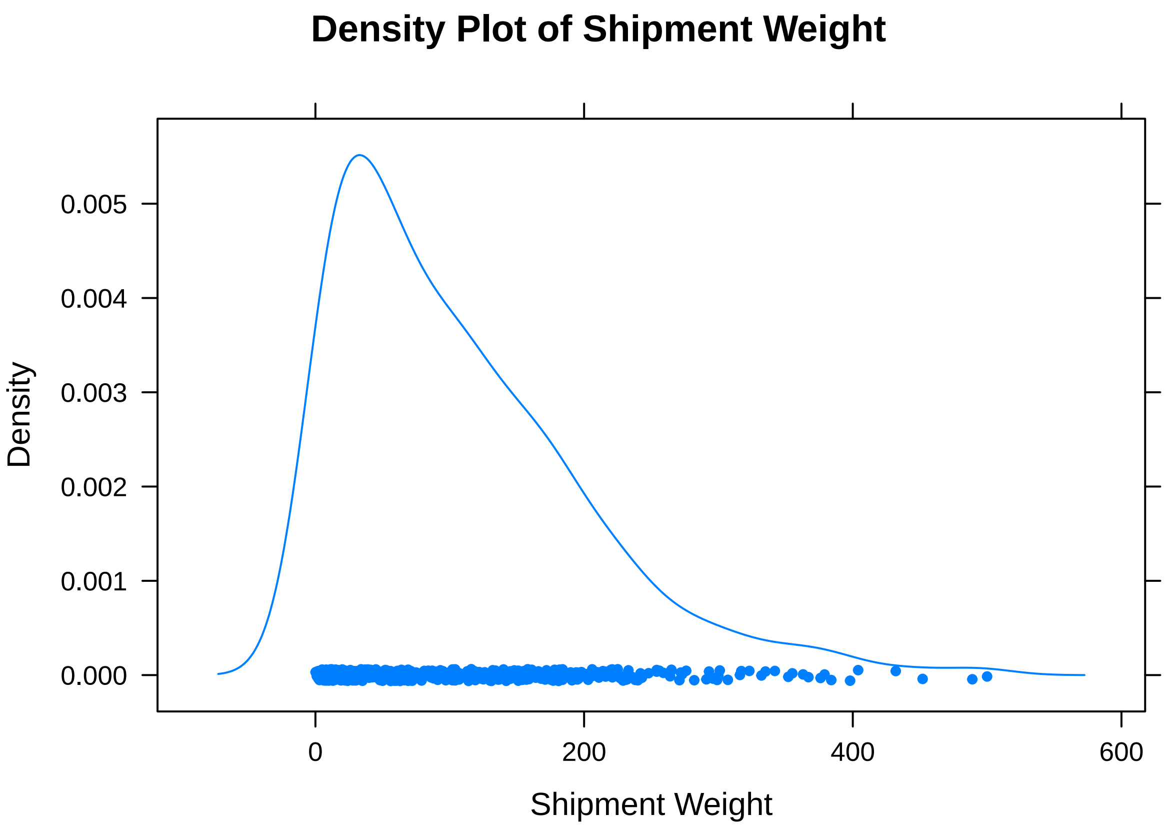
```
densityplot(~ MailOrder_MS$CS_MS, pch = 20,
            main="Density Plot of Past Customer Order",
            xlab="Customer Order")
```



```
densityplot(~ MailOrder_MS$ML_MS, pch = 20,
            main="Density Plot of Delivery Distance",
            xlab="Delivery Distance")
```

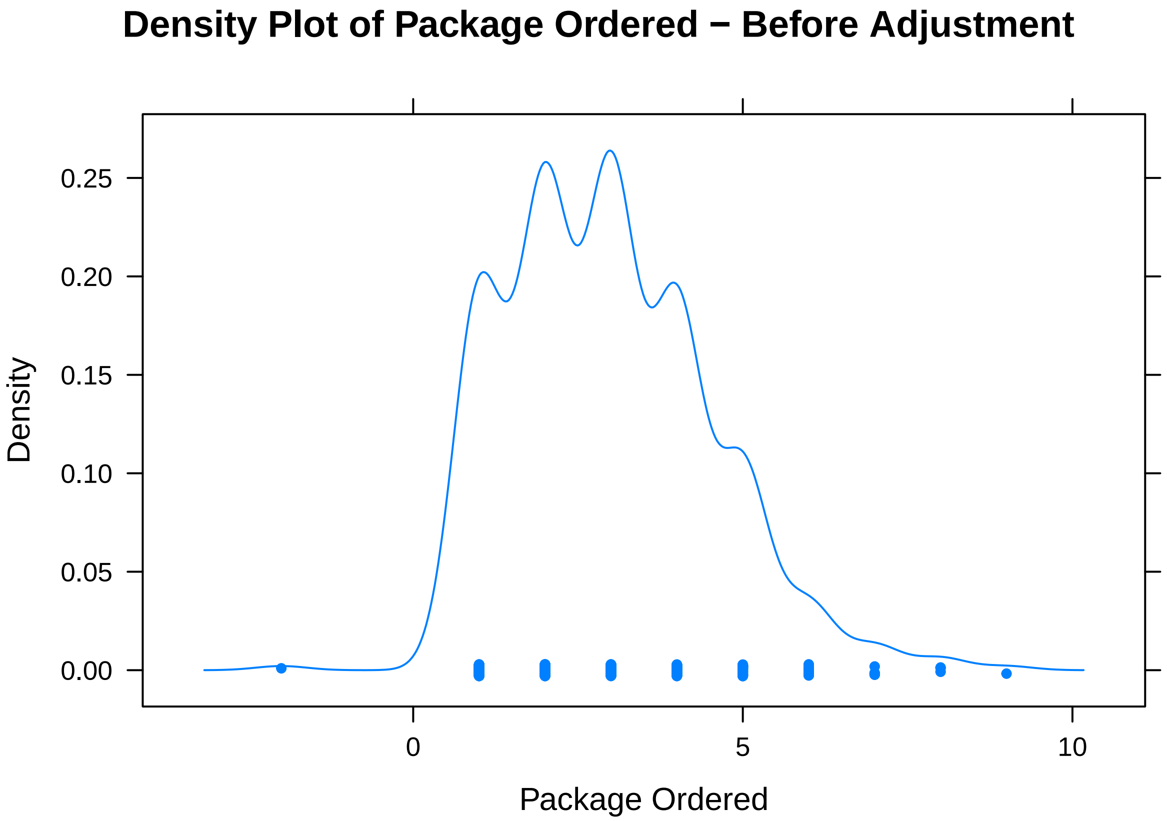


```r
densityplot(~ MailOrder_MS$WT_MS, pch = 20,
            main="Density Plot of Shipment Weight",
            xlab="Shipment Weight")
```

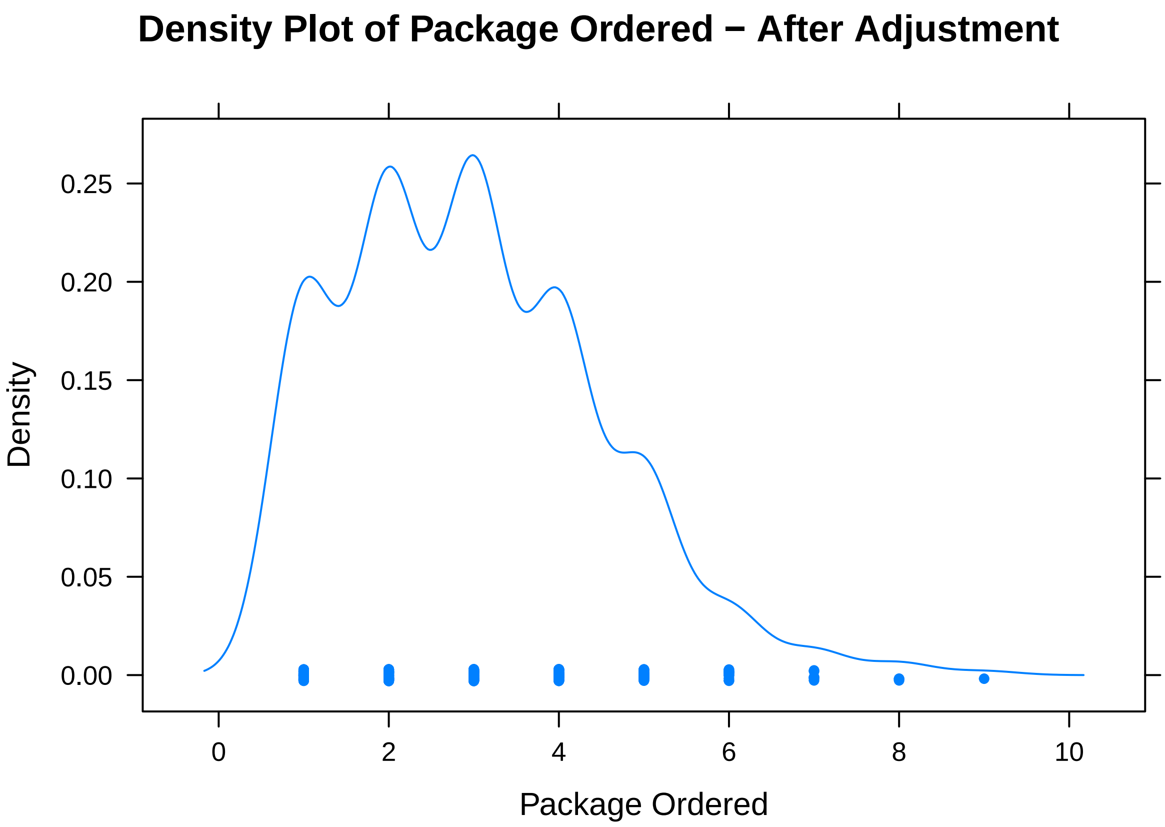


Interpretation: Data looks reasonable except an outlier in Product Order. Removing the Outlier. Density plot before and after removing the outlier.

```
densityplot(~ MailOrder_MS$PG_MS, pch = 20,
            main="Density Plot of Package Ordered - Before Adjustment",
            xlab="Package Ordered")
```



```
nr_ms <- which(MailOrder_MS$PG_MS == min(MailOrder_MS$PG_MS)) #Row number for the minimum value
MailOrder_MS <- MailOrder_MS[-c(nr_ms),]
densityplot(~ MailOrder_MS$PG_MS, pch = 20,
            main="Density Plot of Package Ordered - After Adjustment",
            xlab="Package Ordered")
```



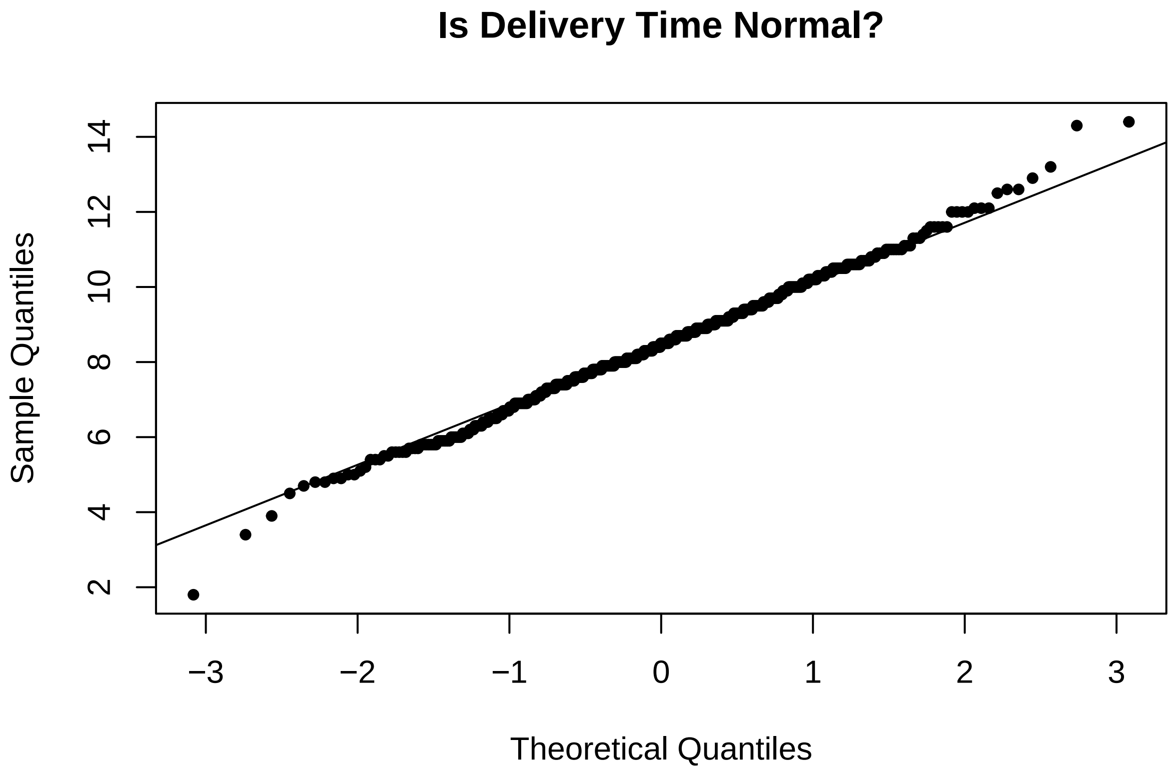
##1.3 Delivery time comparison between Careers Assumptions: Data is Independent Data is normally distributed Variance is unknown, but equal

We can conduct a hypothesis testing to determine if one Carrier has faster delivery times than the other. The flow chart that was discussed in class can help to determine which test to conduct. Since the outcome is continuous we should go for mean comparison test. First we can conduct shapiro test and Q-Q plot to check if data is normal.

```
shapiro.test(MailOrder_MS$DL_MS)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  MailOrder_MS$DL_MS
## W = 0.9964, p-value = 0.3443
```

```
qqnorm(MailOrder_MS$DL_MS, main="Is Delivery Time Normal?", pch=20)
qqline(MailOrder_MS$DL_MS)
```



```
var.test(DL_MS ~ CR_MS, data = MailOrder_MS)
```

```
##
##  F test to compare two variances
##
## data:  DL_MS by CR_MS
## F = 0.92617, num df = 200, denom df = 284, p-value = 0.5633
## alternative hypothesis: true ratio of variances is not equal to 1
## 95 percent confidence interval:
##  0.7187036 1.2004504
## sample estimates:
## ratio of variances
##           0.926171
```

Interpretation: Since p-value from shapiro test is grater than 0.05, also checking the Q-Q normal plot we can accept the hypothesis that distribution is normal.

p-value of grater than 0.05 in F-test confirms that variances are equal.

Above tests meets our assumptions and we can select t-test to check if one Career has faster delivery times than other.

```
t.test(DL_MS ~ CR_MS, data = MailOrder_MS, var.equal = TRUE)
```

```
##
```

```
##  Two Sample t-test
##
## data:  DL_MS by CR_MS
## t = -6.9147, df = 484, p-value = 0.00000000001488
## alternative hypothesis: true difference in means between group Def Post and group Sup Del is not equa
## 95 percent confidence interval:
##  -1.3544364 -0.7550164
## sample estimates:
## mean in group Def Post  mean in group Sup Del
##               7.845274               8.900000
```

Interpretation: Since p-value of t-test is less than 0.05 we can reject the null hypothesis that means are equal and accept the alternate hypothesis, conclude that one Career has faster delivery times than other. We cannot conclude which career is faster than other.

The confidence interval is -1.3544364 and -0.7550164 of mean between 'Def Post' and 'Sup Del'. We can be 95% confident that the true difference in mean between 'Def Post' and 'Sup Del' is within this interval.

##1.4 Split the data set into Training and Test set

```
#Choosing sampling rate for training data
sr_ms <- 0.8 #80% in training set

# Finding the number of rows of data
n.row <- nrow(MailOrder_MS) #counting number of rows

#Choose the rows for the training sample

set.seed(6024) #setting a seed, same starting point. Last 4 digits of my student ID
training.rows <- sample(1:n.row, sr_ms*n.row, replace=FALSE) #sampling
#selecting from 1 to no of rows, how much - sampling-rate*no or rows, placement equal false - don't wan

#Assigning to the training sample
train_ms <- subset(MailOrder_MS[training.rows,]) #creating training data set, only keeping training row

# Assign the balance to the Test Sample

test_ms <- subset(MailOrder_MS[-c(training.rows),]) #keeping everything except training rows

#Checking Train and Test datasets
head(training.rows)
```

```
## [1] 195 305 179 116 157 316
```

```
head(train_ms)
```

```
##     DL_MS VN_MS PG_MS CS_MS ML_MS DM_MS HZ_MS    CR_MS WT_MS
## 196   3.9   346     1     1   938     I     N Def Post   118
## 306   8.6   256     2     8  1009     I     N  Sup Del     2
## 180   6.9   371     2     7   697     C     N  Sup Del    56
## 116  10.0   461     3     4  1243     C     N  Sup Del    90
## 157   9.1   368     3    10   633     I     N Def Post     8
## 317  10.7   345     5    12   196     C     N  Sup Del    81
```

```
head(test_ms)
```

```
##    DL_MS VN_MS PG_MS CS_MS ML_MS DM_MS HZ_MS    CR_MS WT_MS
## 1    8.1   324     5    13   313     C     N  Sup Del   216
## 2    8.4   135     2    13   830     I     N  Sup Del   160
## 5    5.4   321     1     2   221     C     N Def Post    14
## 26   7.9   354     1     5   181     I     N  Sup Del     8
## 29  10.9   357     5    10   684     C     N  Sup Del   130
## 30   7.3   354     1     2   576     C     H  Sup Del    95
```

```
summary(MailOrder_MS)
```

```
##      DL_MS            VN_MS           PG_MS           CS_MS
##  Min.   : 1.800   Min.   : 85.0   Min.   :1.000   Min.   : 0.000
##  1st Qu.: 7.400   1st Qu.:263.0   1st Qu.:2.000   1st Qu.: 5.000
##  Median : 8.500   Median :322.0   Median :3.000   Median : 8.000
##  Mean   : 8.464   Mean   :318.7   Mean   :2.961   Mean   : 9.228
##  3rd Qu.: 9.575   3rd Qu.:371.0   3rd Qu.:4.000   3rd Qu.:13.000
##  Max.   :14.400   Max.   :495.0   Max.   :9.000   Max.   :24.000
##      ML_MS        DM_MS   HZ_MS        CR_MS         WT_MS
##  Min.   :  35.0   C:343   H: 68   Def Post:201   Min.   :  0.1
##  1st Qu.: 444.2   I:143   N:418   Sup Del :285   1st Qu.: 33.0
##  Median : 697.5                                  Median : 86.5
##  Mean   : 754.2                                  Mean   :107.1
##  3rd Qu.:1021.8                                  3rd Qu.:157.8
##  Max.   :1967.0                                  Max.   :500.0
```

```
summary(test_ms)
```

```
##      DL_MS            VN_MS           PG_MS           CS_MS
##  Min.   : 3.400   Min.   :135.0   Min.   :1.000   Min.   : 1.000
##  1st Qu.: 7.300   1st Qu.:271.2   1st Qu.:2.000   1st Qu.: 5.000
##  Median : 8.300   Median :318.0   Median :3.000   Median : 9.000
##  Mean   : 8.468   Mean   :312.1   Mean   :3.184   Mean   : 9.633
##  3rd Qu.: 9.575   3rd Qu.:364.0   3rd Qu.:4.750   3rd Qu.:14.000
##  Max.   :14.300   Max.   :483.0   Max.   :8.000   Max.   :24.000
##      ML_MS        DM_MS  HZ_MS       CR_MS        WT_MS
##  Min.   :  97.0   C:69   H:11   Def Post:30   Min.   :  2.0
##  1st Qu.: 445.5   I:29   N:87   Sup Del :68   1st Qu.: 39.5
##  Median : 717.0                               Median : 99.5
##  Mean   : 755.9                               Mean   :116.8
##  3rd Qu.:1033.0                               3rd Qu.:168.8
##  Max.   :1807.0                               Max.   :452.0
```

```
summary(train_ms)
```

```
##      DL_MS            VN_MS           PG_MS           CS_MS
##  Min.   : 1.800   Min.   : 85.0   Min.   :1.000   Min.   : 0.000
##  1st Qu.: 7.400   1st Qu.:261.5   1st Qu.:2.000   1st Qu.: 5.000
##  Median : 8.500   Median :324.0   Median :3.000   Median : 8.000
##  Mean   : 8.463   Mean   :320.3   Mean   :2.905   Mean   : 9.126
##  3rd Qu.: 9.525   3rd Qu.:373.2   3rd Qu.:4.000   3rd Qu.:13.000
##  Max.   :14.400   Max.   :495.0   Max.   :9.000   Max.   :24.000
##      ML_MS        DM_MS   HZ_MS        CR_MS           WT_MS
##  Min.   :  35.0   C:274   H: 57   Def Post:171   Min.   :  0.10
##  1st Qu.: 442.0   I:114   N:331   Sup Del :217   1st Qu.: 31.75
##  Median : 695.5                                  Median : 83.50
##  Mean   : 753.7                                  Mean   :104.67
##  3rd Qu.:1012.2                                  3rd Qu.:151.25
##  Max.   :1967.0                                  Max.   :500.00
```

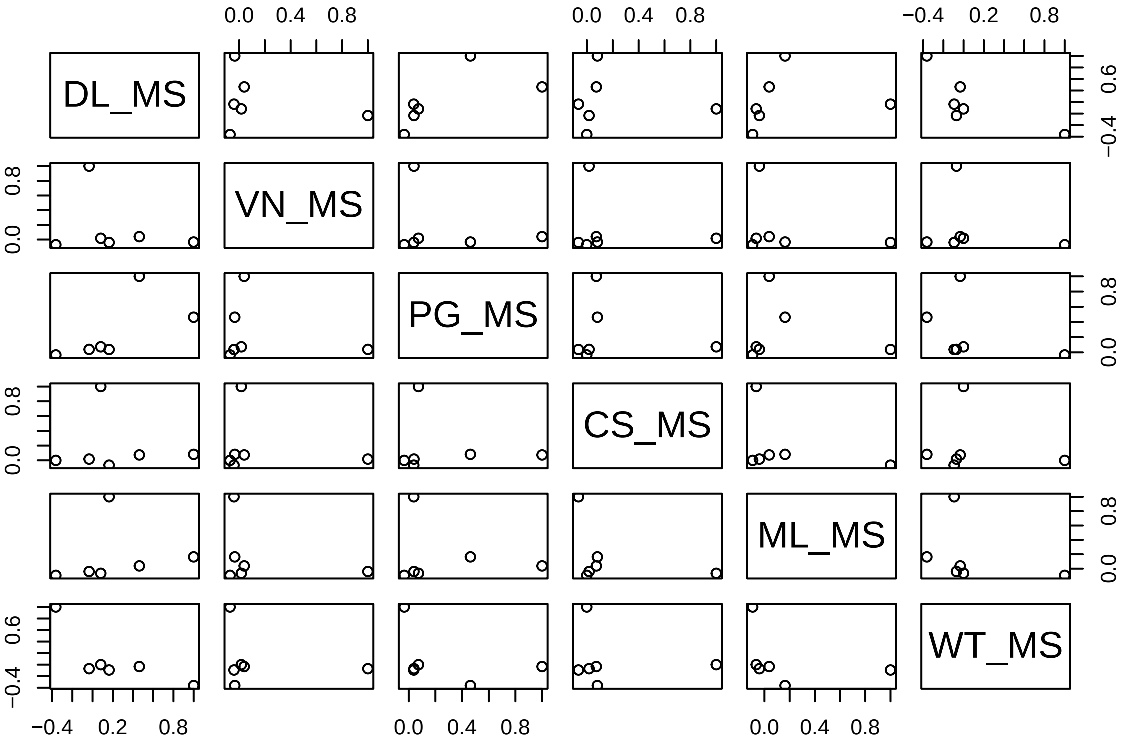
Part 2 - Simple Linear Regression

##2.1 - Correlations Using correlation matrix and corrgram to determine correlation between numeric variables.

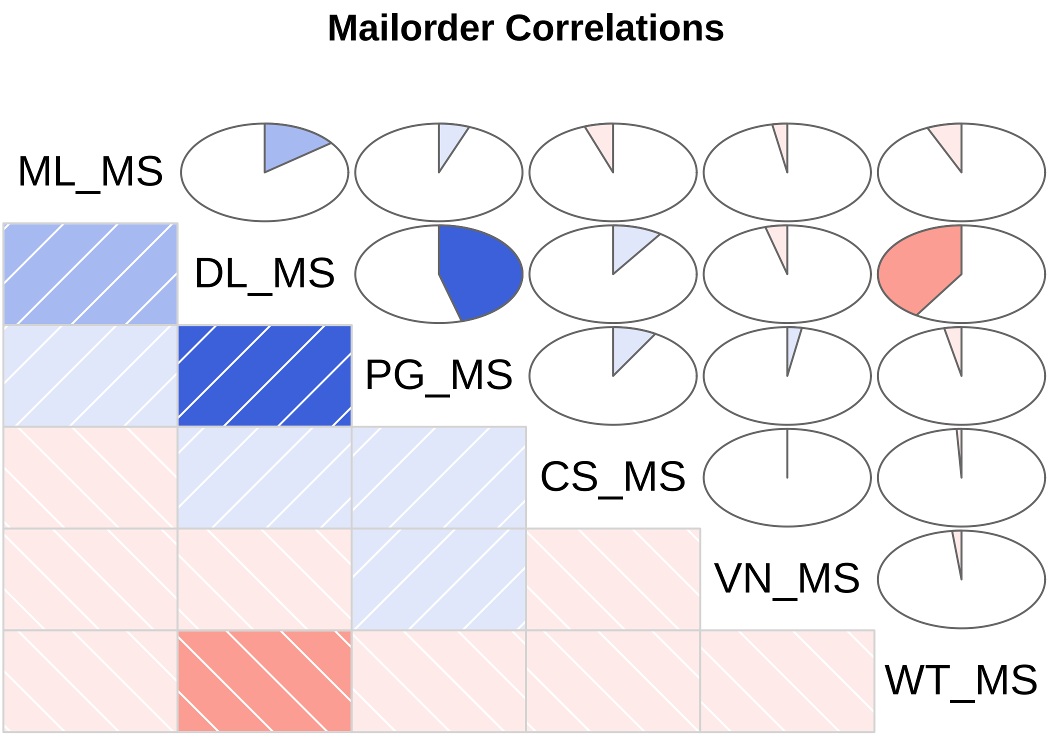
```
#Numeric Correlation
trnCr_MS <- cor(train_ms[-c(6:8)], method="spearman") #Excluding non numeric columns for correlation
round(trnCr_MS, 2)
```

```
##       DL_MS VN_MS PG_MS CS_MS ML_MS WT_MS
## DL_MS  1.00 -0.03  0.46  0.08  0.16 -0.36
## VN_MS -0.03  1.00  0.04  0.02 -0.04 -0.07
## PG_MS  0.46  0.04  1.00  0.07  0.04 -0.03
## CS_MS  0.08  0.02  0.07  1.00 -0.07  0.00
## ML_MS  0.16 -0.04  0.04 -0.07  1.00 -0.09
## WT_MS -0.36 -0.07 -0.03  0.00 -0.09  1.00
```

```
#Graphical Correlation
pairs(trnCr_MS)
```



```
corrgram(train_ms, order=TRUE, lower.panel=panel.shade,
         upper.panel=panel.pie, text.panel=panel.txt,
         main="Mailorder Correlations")
```



Interpretation: Reviewing the correlation matrix and corregram we can conclude that there is moderate positive correlation between Time for Delivery and Packages of product ordered. This make sense probably larger order takes priority and delivered faster.

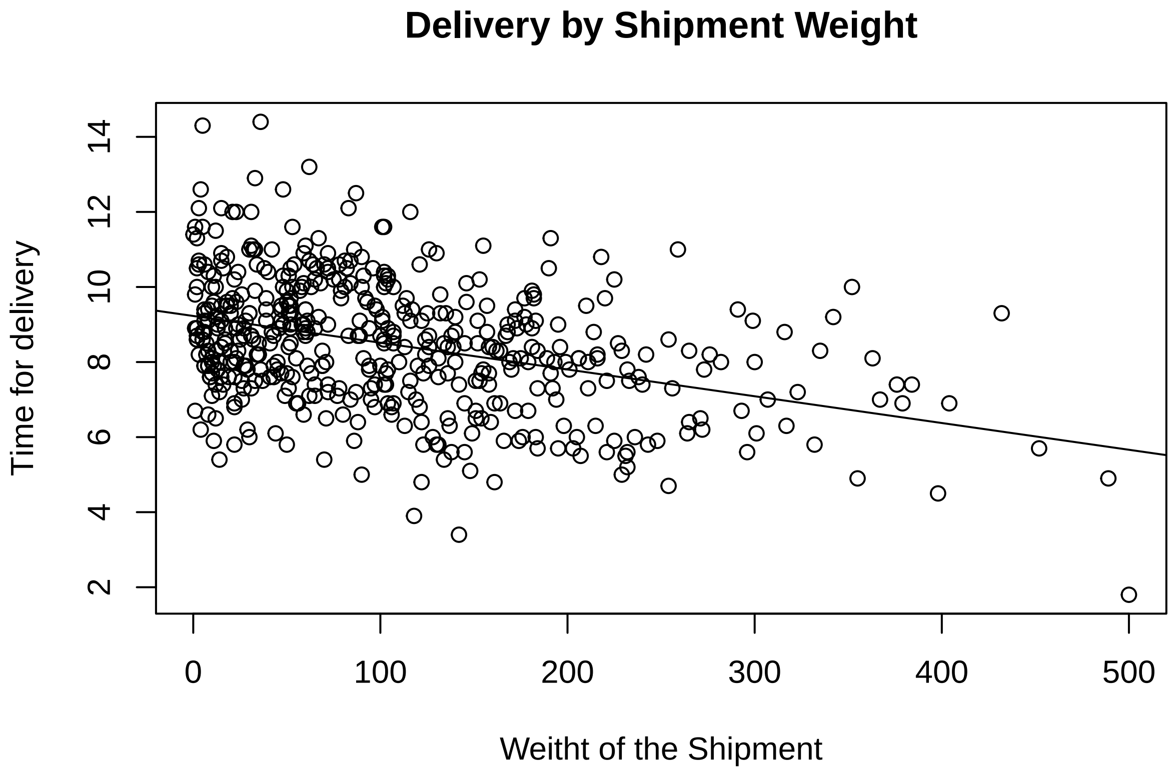
There is also moderate negative correlation between Time for Delivery and Weight of the shipment. This also make sense as heavy weight shipment takes more processing time and increases delivery time.

##2.2 Creating a simple linear regression model using time for delivery as the dependent variable and weight of the shipment as the independent.

```
MailorderModel1_MS <- lm(DL_MS ~ WT_MS, data=MailOrder_MS) #Model 1
MailorderModel1_MS
```

```
##
## Call:
## lm(formula = DL_MS ~ WT_MS, data = MailOrder_MS)
##
## Coefficients:
## (Intercept)        WT_MS
##     9.22757     -0.00713
```

```
plot(DL_MS ~ WT_MS, data=MailOrder_MS,
     main="Delivery by Shipment Weight",
     xlab = "Weitht of the Shipment",
     ylab = "Time for delivery")
abline(MailorderModel1_MS)
```



```
summary(MailorderModel1_MS)
```

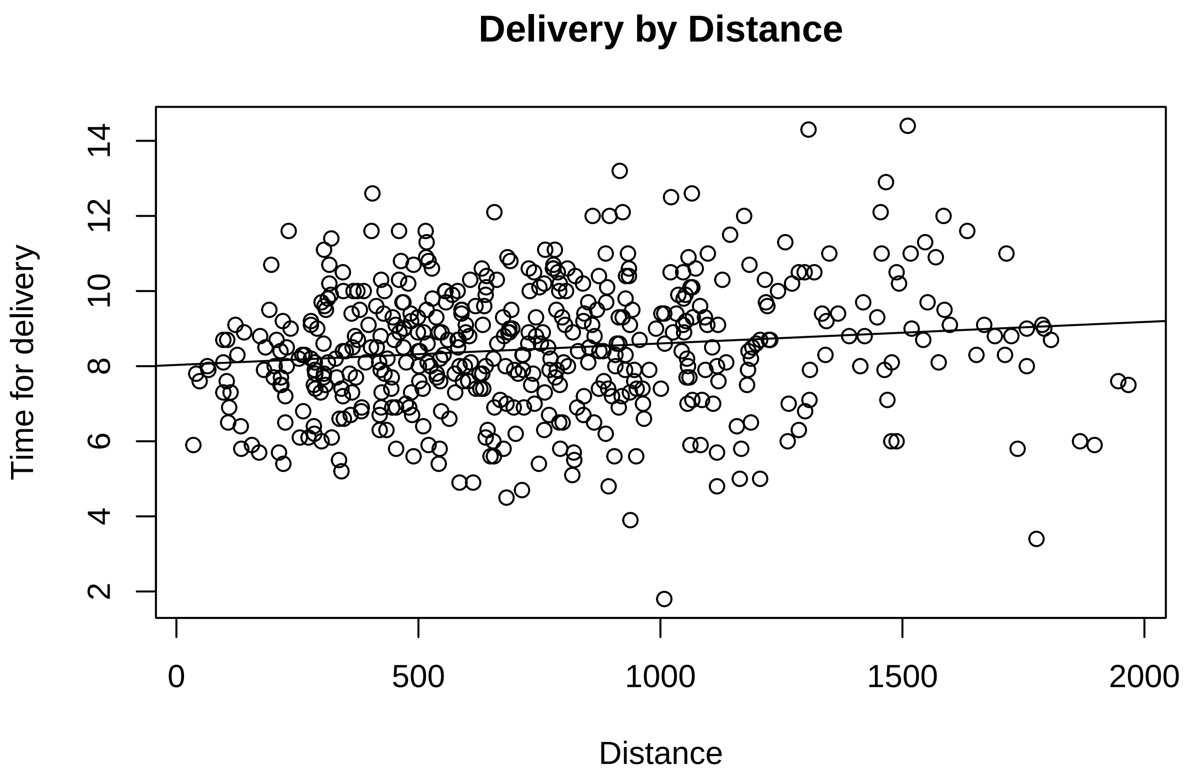
```
##
## Call:
## lm(formula = DL_MS ~ WT_MS, data = MailOrder_MS)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.8151 -1.1775  0.0611  1.0769  5.4291
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  9.2275747  0.1113571  82.865   <2e-16 ***
## WT_MS       -0.0071301  0.0007866  -9.065   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.605 on 484 degrees of freedom
## Multiple R-squared:  0.1451, Adjusted R-squared:  0.1434
## F-statistic: 82.17 on 1 and 484 DF,  p-value: < 2.2e-16
```

##2.3 Creating a simple linear regression model using time for delivery as the dependent variable and distance the shipment needs to travel as the independent.

```
MailorderModel2_MS <- lm(DL_MS ~ ML_MS, data=MailOrder_MS) #Model 2
MailorderModel2_MS
```

```
##
## Call:
## lm(formula = DL_MS ~ ML_MS, data = MailOrder_MS)
##
## Coefficients:
## (Intercept)        ML_MS
##   8.0323493    0.0005721
```

```
plot(DL_MS ~ ML_MS, data=MailOrder_MS,
     main="Delivery by Distance",
     xlab = "Distance",
     ylab = "Time for delivery")
abline(MailorderModel2_MS)
```



```
summary(MailorderModel2_MS)
```

```
##
## Call:
## lm(formula = DL_MS ~ ML_MS, data = MailOrder_MS)
##
## Residuals:
```

```
##     Min      1Q  Median      3Q     Max
## -6.8090 -1.0061 -0.0249  1.1524  5.5205
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 8.0323493  0.1632134  49.214  < 2e-16 ***
## ML_MS       0.0005721  0.0001901   3.009  0.00275 **
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
##
## Residual standard error: 1.72 on 484 degrees of freedom
## Multiple R-squared:  0.01837,    Adjusted R-squared:  0.01634
## F-statistic: 9.057 on 1 and 484 DF,  p-value: 0.002754
```

```r
### Comparing the RMSE
#Model 1 RMSE
print("Model 1 RMSE: ")
```

```
## [1] "Model 1 RMSE: "
```

```r
M1pred_MS <- predict(MailorderModel1_MS, newdata=train_ms)

RMSE_trnM1_MS <- sqrt(mean((MailOrder_MS$WT_MS - M1pred_MS)^2))
round(RMSE_trnM1_MS,3)
```

```
## [1] 135.266
```

```r
M1pred_MS <- predict(MailorderModel1_MS, newdata=test_ms)

RMSE_tstM1_MS <- sqrt(mean((MailOrder_MS$WT_MS - M1pred_MS)^2))
round(RMSE_tstM1_MS,3)
```

```
## [1] 135.317
```

```r
#Model 2 RMSE
print("Model 2 RMSE: ")
```

```
## [1] "Model 2 RMSE: "
```

```r
M1pred_MS <- predict(MailorderModel2_MS, newdata=train_ms)

RMSE_trnM1_MS <- sqrt(mean((MailOrder_MS$WT_MS - M1pred_MS)^2))
round(RMSE_trnM1_MS,3)
```

```
## [1] 135.284
```

```r
M1pred_MS <- predict(MailorderModel2_MS, newdata=test_ms)

RMSE_tstM1_MS <- sqrt(mean((MailOrder_MS$WT_MS - M1pred_MS)^2))
round(RMSE_tstM1_MS,3)
```

```
## [1] 135.278
```

##2.4 Model comparison

Comparing following 5 measures for both models.

a. F-Stat - p-value for both model is less than 0.05
b. Adjusted R-squared - comparing values for both model (0.1434 and 0.01634), model1 seems better
c. Residuals for both models are centered around 0 and symmetric
d. t-test value for both WT and ML is less than 0.05, so both passed
e. Coefficient - looks reasonable
f. RMSE are almost same for both models

Based on above comparison I conclude that both models are not good enough. Just based on Adjusted R-squared, Model1 is superior than Model2.

Part 3 - Multiple Linear Regression

Creating a full model using all the variables.

```
full.model_ms = lm(DL_MS ~ . , data=train_ms, na.action=na.omit) #. means every other variable

summary(full.model_ms )
```

```
##
## Call:
## lm(formula = DL_MS ~ ., data = train_ms, na.action = na.omit)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.1301 -0.7120  0.0093  0.7661  4.0405
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)   7.4848627  0.3801939  19.687  < 2e-16 ***
## VN_MS        -0.0009468  0.0008444  -1.121 0.262880
## PG_MS         0.5390913  0.0425553  12.668  < 2e-16 ***
## CS_MS         0.0190336  0.0121027   1.573 0.116629
## ML_MS         0.0003465  0.0001517   2.285 0.022880 *
## DM_MSI        0.4889559  0.1353854   3.612 0.000345 ***
## HZ_MSN       -0.8843005  0.1745228  -5.067 6.33e-07 ***
## CR_MSSup Del  1.0056338  0.1258981   7.988 1.67e-14 ***
## WT_MS        -0.0064149  0.0006835  -9.386  < 2e-16 ***
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
##
## Residual standard error: 1.207 on 379 degrees of freedom
## Multiple R-squared:  0.5069, Adjusted R-squared:  0.4964
## F-statistic: 48.69 on 8 and 379 DF,  p-value: < 2.2e-16
```

```
pred_ms <- predict(full.model_ms , newdata=train_ms)

RMSE_trn_full_ms <- sqrt(mean((train_ms$DL_MS - pred_ms)^2))
round(RMSE_trn_full_ms,2)
```

```
## [1] 1.19
```

```
RMSE_tst_full_ms <- sqrt(mean((test_ms$DL_MS - pred_ms)^2))
round(RMSE_tst_full_ms,2)
```

```
## [1] 2.17
```

Interpretation: Comparing following 5 measures for full model.

a. F-Stat - p-value for full model is less than 0.05.
b. Adjusted R-squared - value of 0.4964 indicate a moderate model not excellent.
c. Residuals - Residuals are centered around 0 and symmetric.
d. t-test - 5/8 variable has less than 0.05 t value, so 5 variable out of 8 passed t-test.
e. Coefficient - coefficients looks reasonable
f. RMSE - train RMSE of 1.19 is less than test RMSE of 2.16, so this model is over fitted.

Creating a model using backward selection.

```
back.model_ms = step(full.model_ms, direction="backward", details=TRUE)
```

```
## Start:  AIC=154.7
## DL_MS ~ VN_MS + PG_MS + CS_MS + ML_MS + DM_MS + HZ_MS + CR_MS +
##     WT_MS
##
##         Df Sum of Sq    RSS    AIC
## - VN_MS  1     1.831 553.71 153.99
## <none>               551.88 154.70
## - CS_MS  1     3.601 555.48 155.23
## - ML_MS  1     7.601 559.48 158.01
## - DM_MS  1    18.993 570.88 165.83
## - HZ_MS  1    37.385 589.27 178.13
## - CR_MS  1    92.907 644.79 213.07
## - WT_MS  1   128.269 680.15 233.79
## - PG_MS  1   233.682 785.56 289.69
##
## Step:  AIC=153.99
## DL_MS ~ PG_MS + CS_MS + ML_MS + DM_MS + HZ_MS + CR_MS + WT_MS
##
##         Df Sum of Sq    RSS    AIC
## <none>               553.71 153.99
## - CS_MS  1     3.670 557.38 154.55
## - ML_MS  1     7.794 561.51 157.41
## - DM_MS  1    18.386 572.10 164.66
## - HZ_MS  1    38.481 592.19 178.06
## - CR_MS  1    94.768 648.48 213.29
## - WT_MS  1   127.418 681.13 232.35
## - PG_MS  1   232.895 786.61 288.21
```

```r
summary(back.model_ms)
```

```
##
## Call:
## lm(formula = DL_MS ~ PG_MS + CS_MS + ML_MS + DM_MS + HZ_MS +
##     CR_MS + WT_MS, data = train_ms, na.action = na.omit)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.1915 -0.7115 -0.0003  0.7451  3.9725
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)   7.1848079  0.2701599  26.595  < 2e-16 ***
## PG_MS         0.5380573  0.0425597  12.642  < 2e-16 ***
## CS_MS         0.0192121  0.0121057   1.587  0.11334
## ML_MS         0.0003508  0.0001517   2.313  0.02127 *
## DM_MSI        0.4802930  0.1352105   3.552  0.00043 ***
## HZ_MSN       -0.8956570  0.1742876  -5.139 4.43e-07 ***
## CR_MSSup Del  1.0139081  0.1257242   8.065 9.70e-15 ***
## WT_MS        -0.0063903  0.0006834  -9.351  < 2e-16 ***
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
##
## Residual standard error: 1.207 on 380 degrees of freedom
## Multiple R-squared:  0.5052, Adjusted R-squared:  0.4961
## F-statistic: 55.43 on 7 and 380 DF,  p-value: < 2.2e-16
```

```r
pred_ms <- predict(back.model_ms, newdata=train_ms)

RMSE_trn_back_ms <- sqrt(mean((train_ms$DL_MS - pred_ms)^2))
round(RMSE_trn_back_ms,2)
```

```
## [1] 1.19
```

```r
RMSE_tst_back_ms <- sqrt(mean((test_ms$DL_MS - pred_ms)^2))
round(RMSE_tst_back_ms,2)
```

```
## [1] 2.16
```

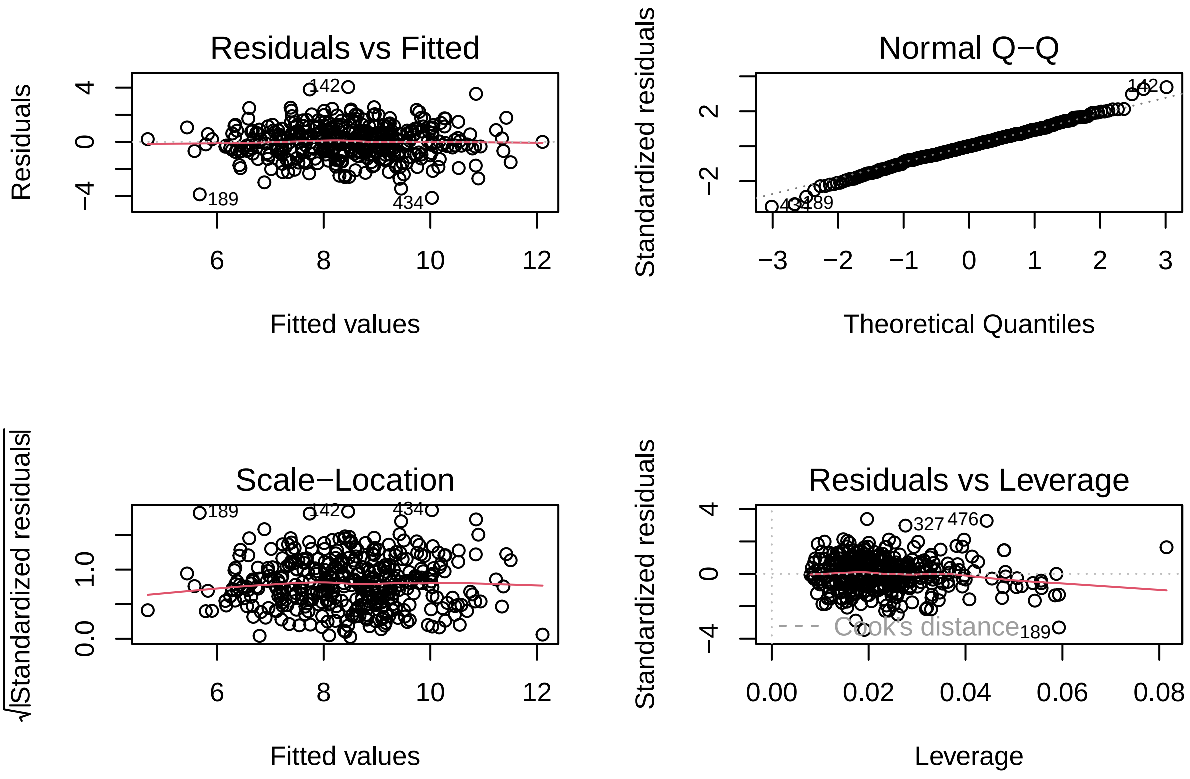
Interpretation: Comparing following 5 measures for backward model.

a. F-Stat - p-value for backward selection model is less than 0.05.
b. Adjusted R-squared - value of 0.4961 indicate a moderate model not excellent.
c. Residuals - Residuals are centered around 0 and symmetric.
d. t-test - 5/7 variable has less than 0.05 t value, so 5 variable out of 7 passed t-test.
e. Coefficient - Coefficients looks reasonable
f. RMSE - train RMSE of 1.19 is less than test RMSE of 2.16, so this model is over fitted.

Part 4 - Model Evaluation

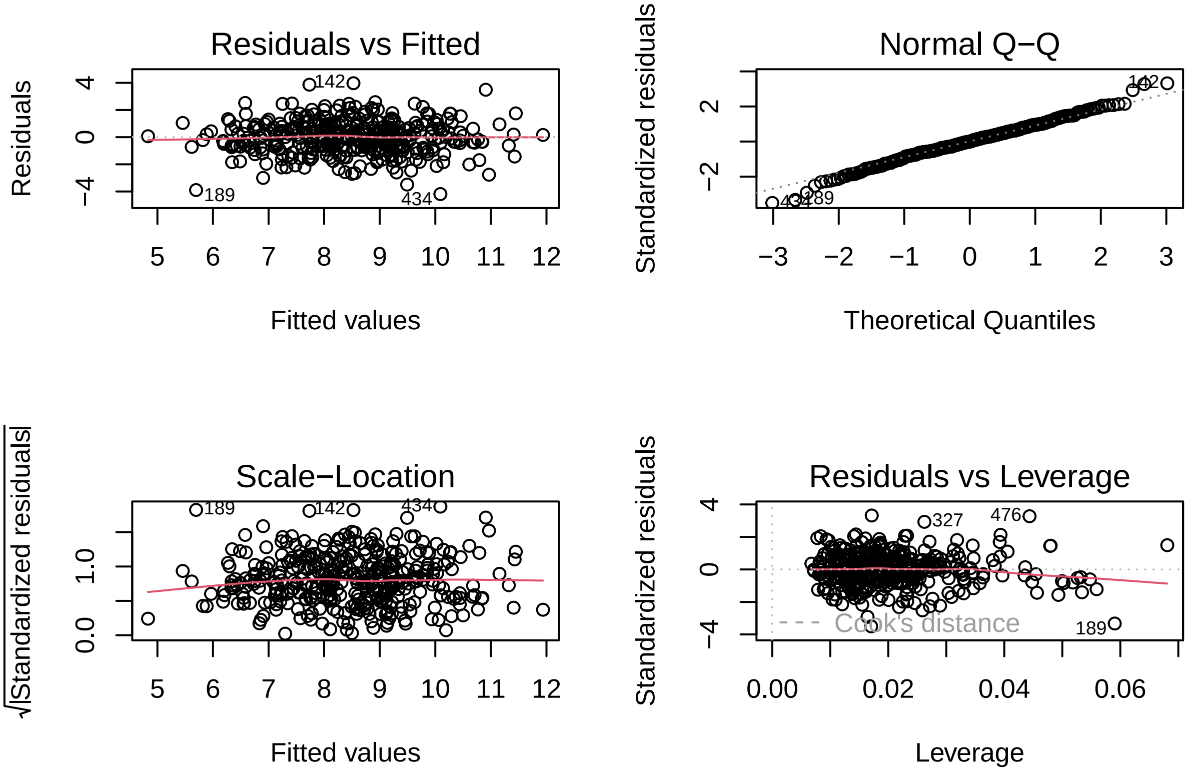
Evaluating main assumptions of regression. Checking for Error Terms, Constant variance, normal distribution and independence of predictors

```
#Checking for Linearity
par(mfrow = c(2, 2))
plot(full.model_ms)
```



```
par(mfrow = c(1, 1))

par(mfrow = c(2, 2))
plot(back.model_ms)
```



```
par(mfrow = c(1, 1))

#Checking for Normality
full.res_ms <- residuals(full.model_ms)
back.res_ms <- residuals(back.model_ms)

shapiro.test(full.res_ms)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  full.res_ms
## W = 0.99431, p-value = 0.1581
```

```
shapiro.test(back.res_ms)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  back.res_ms
## W = 0.9942, p-value = 0.1477
```

Interpretation: Evaluating main assumptions of regression:

Linearity: The residuals are evenly distributed around zero for both models. Linearity assumption is met.

Normality: Reviewing the Q-Q plot we see the residuals are approximately normally distributed for both models.Normality assumption is met.

Homoscedasticity: Reviewing the residual vs fitted plot of the residuals against the predicted values for both models we see that the residuals are evenly distributed around zero and there is no clear pattern in the plot, the assumption is met.

Independence: Observing the Scale-Location graph for both models we see there is no clear pattern in the plot, the assumption is met.

Resuduals vs Leverage: Within Cook's distance with high leverage and low influence for both models. No significant influential value.

Part 5 - Final Recommendation

Two models created in part 3 has similar test results for all tests except t-test. For full model 5/8 variables passed t-test on the other hand 5/7 variable passed for backward selection model. Backward selection model is providing more accurate result using less variables. So we can suggest that backward model is superior than full model.