Prediction

Divyang Vinubhai Hirpara

13/03/2023

1. Preliminary and Exploratory

1. Rename all variables with your initials appended

```
data_DVH <- read.table("PROG8430_Assign_dataset.txt",sep=",",header = TRUE)
head(data_DVH)

##      DL  VN PG CS   ML DM HZ       CR  WT
## 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
## 6   9.4 397  2  8 1002  I  N  Sup Del  47
```

Interpretation

- The text(.txt) file shall be read with 'read.table' function in R.
- Text file is comma separated hence, sep=" ," is used to identify a rows and column.
- Header=TRUE is used due to the text file is generated with header in first line.
- By default, 6 records are displayed with 'head()' function as shown above.
- There are total 9 columns of DL,VN, PG, CS, ML, DM, HZ, CR and WT with different datatype.

Rename Variables of column name

```
#Append data_DVH initials to column names
colnames(data_DVH) <- paste(colnames(data_DVH), "DVH", sep = "_")
head(data_DVH)

##   DL_DVH VN_DVH PG_DVH CS_DVH ML_DVH DM_DVH HZ_DVH   CR_DVH WT_DVH
## 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
## 6    9.4    397      2      8   1002      I      N  Sup Del     47
```

Interpretation

Every column are replaced with initials.

DL –> DL_DVH
VN –> VN_DVH
PG –> PG_DVH
CS –> CS_DVH
ML –> ML_DVH
DM –> DM_DH
HZ –> HZ_DH
CR –> CR_DVH
WT –> WT_DVH

2. Examine the data using the exploratory techniques we have learned in class. Does the data look reasonable? Are there any outliers? If so, deal with them appropriately.

Find Missing Value if any

```
summary(data_DVH)
```

```
##      DL_DVH           VN_DVH          PG_DVH           CS_DVH
##  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_DVH          DM_DVH             HZ_DVH             CR_DVH
##  Min.   :  35.0   Length:487         Length:487         Length:487
##  1st Qu.: 444.5   Class :character   Class :character   Class :character
##  Median : 697.0   Mode  :character   Mode  :character   Mode  :character
##  Mean   : 754.0
##  3rd Qu.:1021.5
##  Max.   :1967.0
##      WT_DVH
##  Min.   :  0.1
##  1st Qu.: 33.0
##  Median : 87.0
##  Mean   :107.1
##  3rd Qu.:157.5
##  Max.   :500.0
```

Interpretation

Looking at the above summary table of all columns, it seems there is no missing value available in any column.
If any missing value is available in any column, it is supposed to look like this - NA's 2. where 2 represents the number of missing values.

Look for coefficient of Variance

```
stat.desc(data_DVH)  #Consider coef of var
```

```
##                    DL_DVH        VN_DVH       PG_DVH       CS_DVH
## 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_DVH DM_DVH HZ_DVH CR_DVH       WT_DVH
## 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

From the above stat values, it seems there is not likely very low value of Coef.var.

Look for Correlation Filter between variables

```
numeric_data_DVH <- data_DVH[-c(6:8)]
cor(numeric_data_DVH,method="spearman")
```

```
##             DL_DVH       VN_DVH      PG_DVH       CS_DVH      ML_DVH
## DL_DVH  1.00000000 -0.027261928  0.46130819  0.089088635  0.15222013
## VN_DVH -0.02726193  1.000000000  0.01981268 -0.009450598 -0.02177818
## PG_DVH  0.46130819  0.019812684  1.00000000  0.056490697  0.03155844
## CS_DVH  0.08908863 -0.009450598  0.05649070  1.000000000 -0.04566593
## ML_DVH  0.15222013 -0.021778183  0.03155844 -0.045665931  1.00000000
## WT_DVH -0.33178770 -0.029531300 -0.00410228 -0.020432083 -0.05345075
##             WT_DVH
## DL_DVH -0.33178770
## VN_DVH -0.02953130
## PG_DVH -0.00410228
## CS_DVH -0.02043208
```

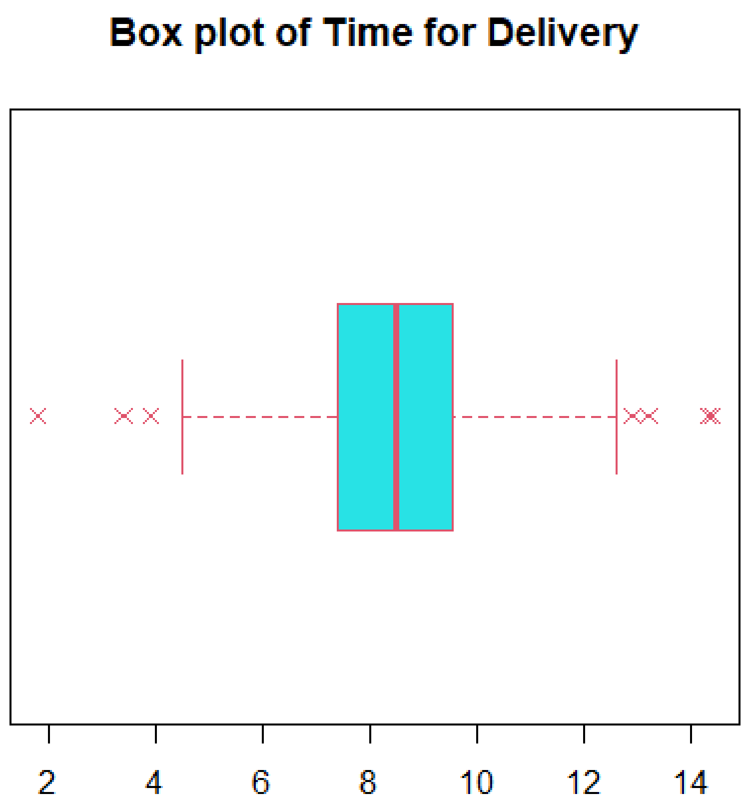
```
## ML_DVH -0.05345075
## WT_DVH  1.00000000
```

Interpretation
With Correlation function cor(), method="spearman" basically it refers to calculation of the Spearman's rank correlation coefficient. It helps find the high correlation between two variable.
From above values, there are no variables available with high correlation value.

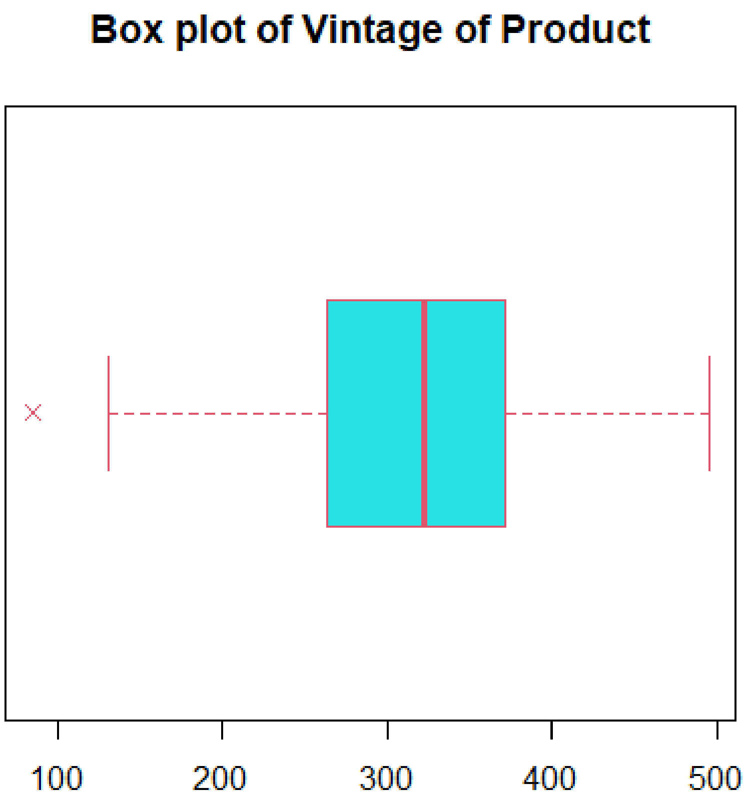
Hence, there is no need to drop any variables.

Look for outliers with Box Plot

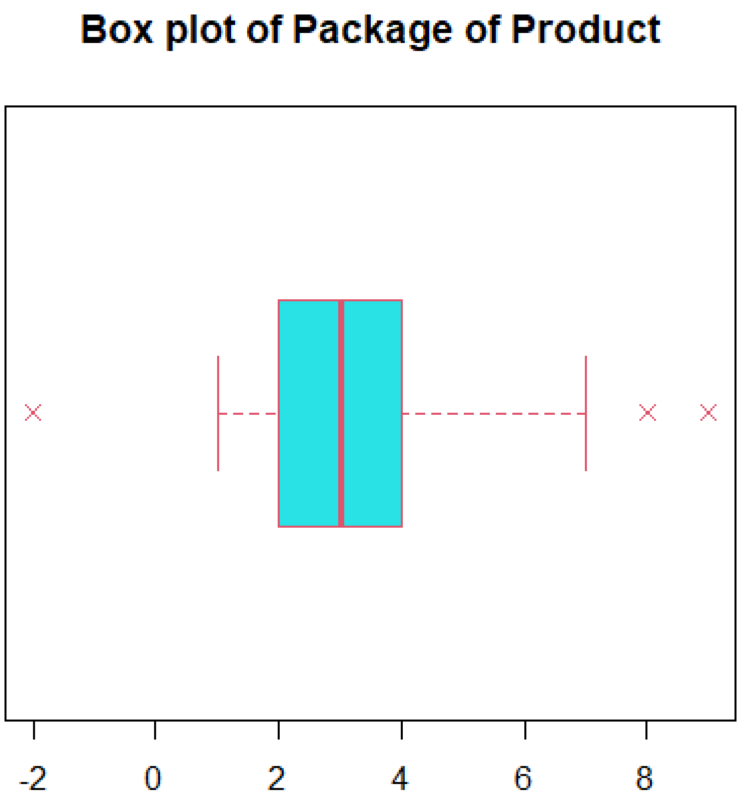
```
boxplot(data_DVH$DL_DVH, horizontal=TRUE, pch=4, col=5, border = 2, main="Box plot of Time for Delivery")
```



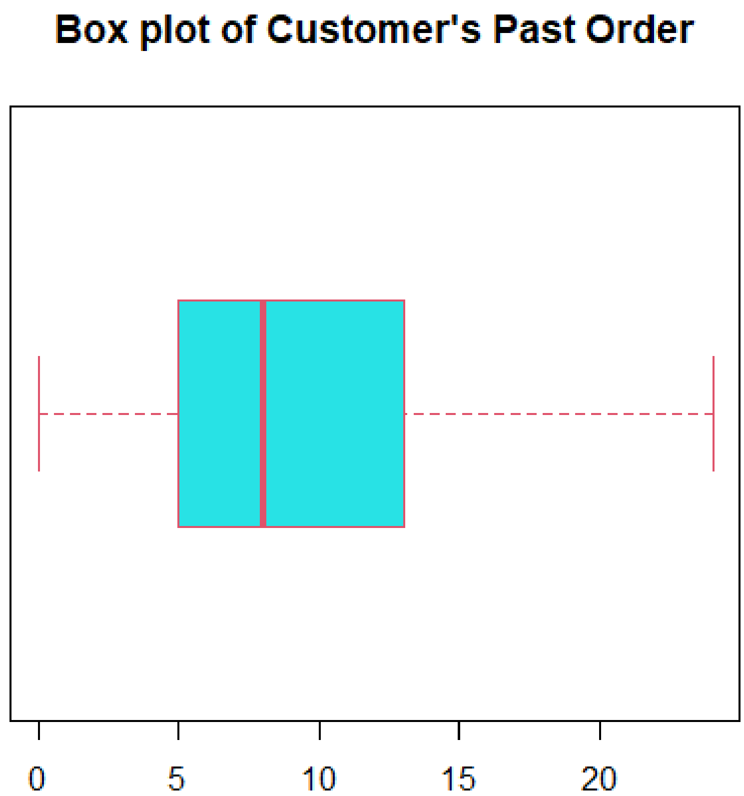

```
boxplot(data_DVH$VN_DVH, horizontal=TRUE, pch=4,col=5, border = 2, main="Box plot of Vintage of Product")
```



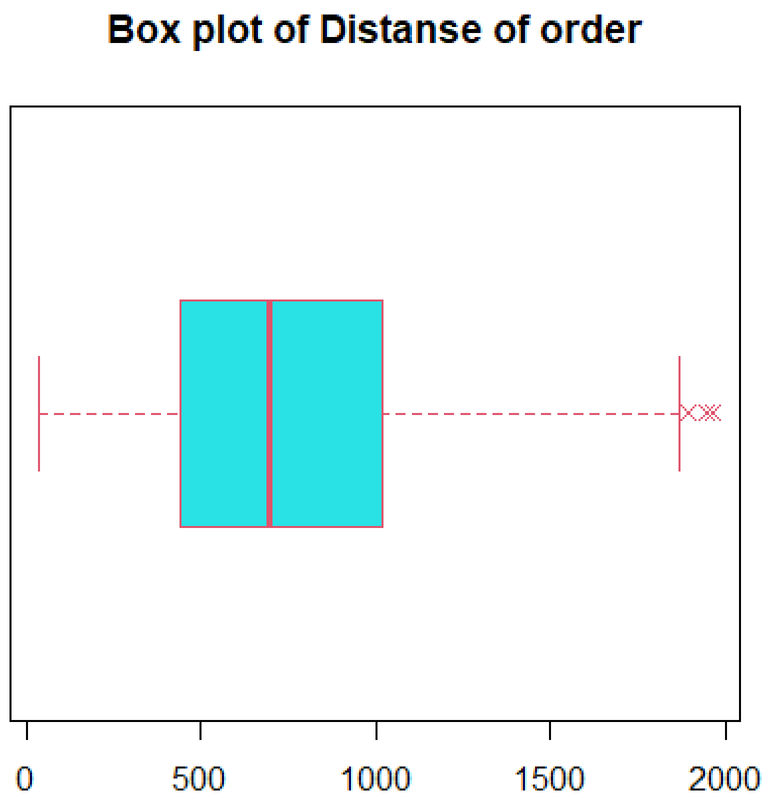

```
boxplot(data_DVH$PG_DVH, horizontal=TRUE, pch=4,col=5, border = 2, main="Box
plot of Package of Product")
```



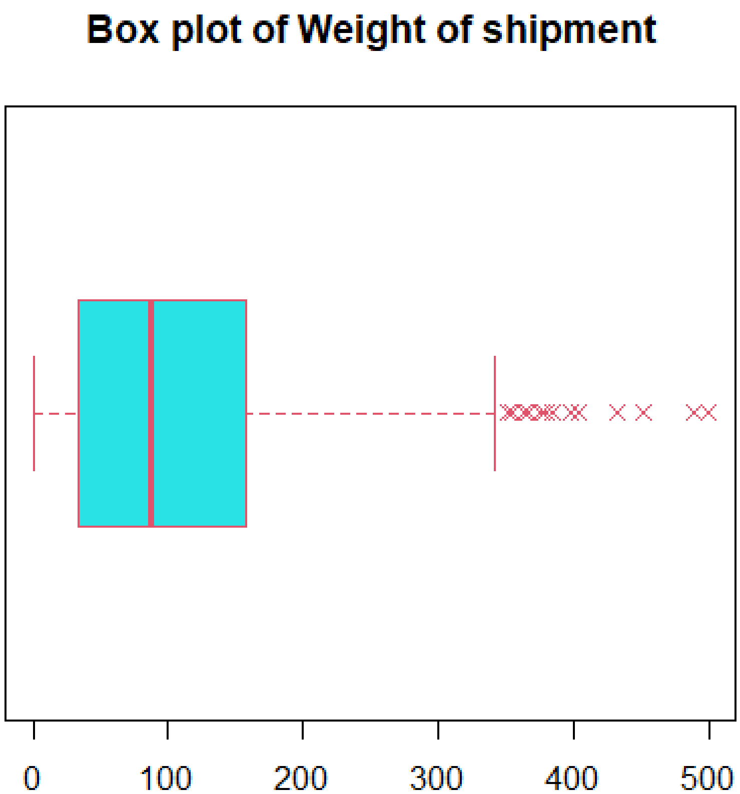

```
boxplot(data_DVH$CS_DVH, horizontal=TRUE, pch=4,col=5, border = 2, main="Box
plot of Customer's Past Order")
```



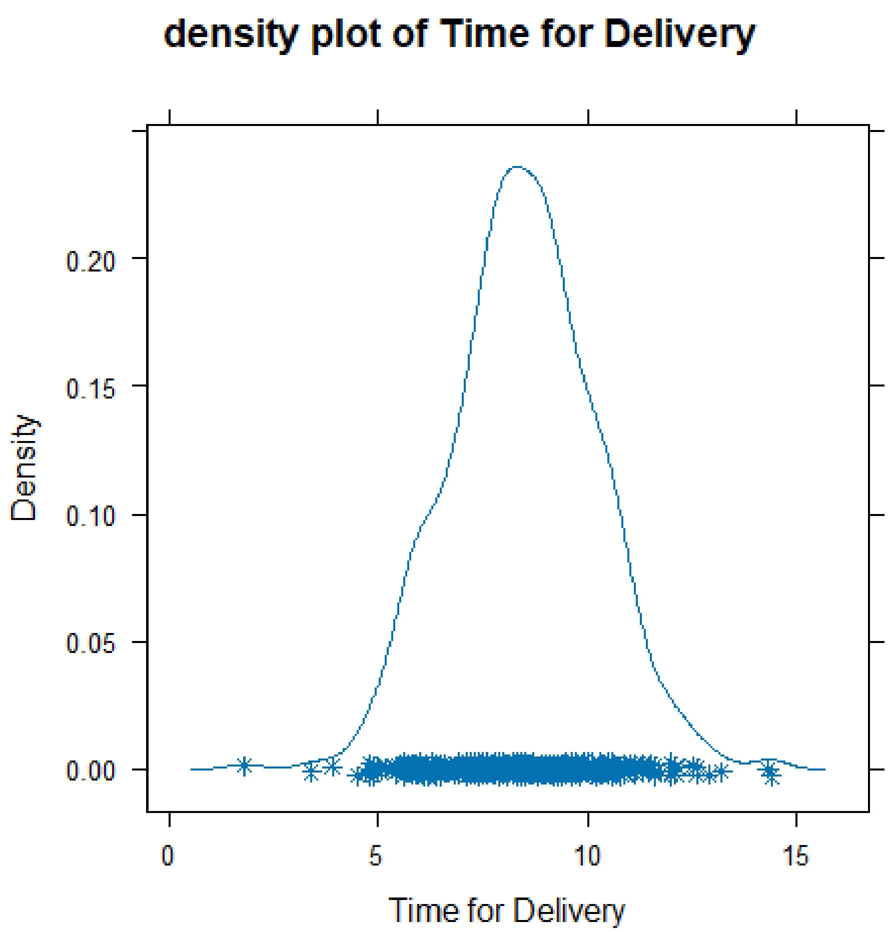

```
boxplot(data_DVH$ML_DVH, horizontal=TRUE, pch=4,col=5, border = 2, main="Box
plot of Distanse of order")
```



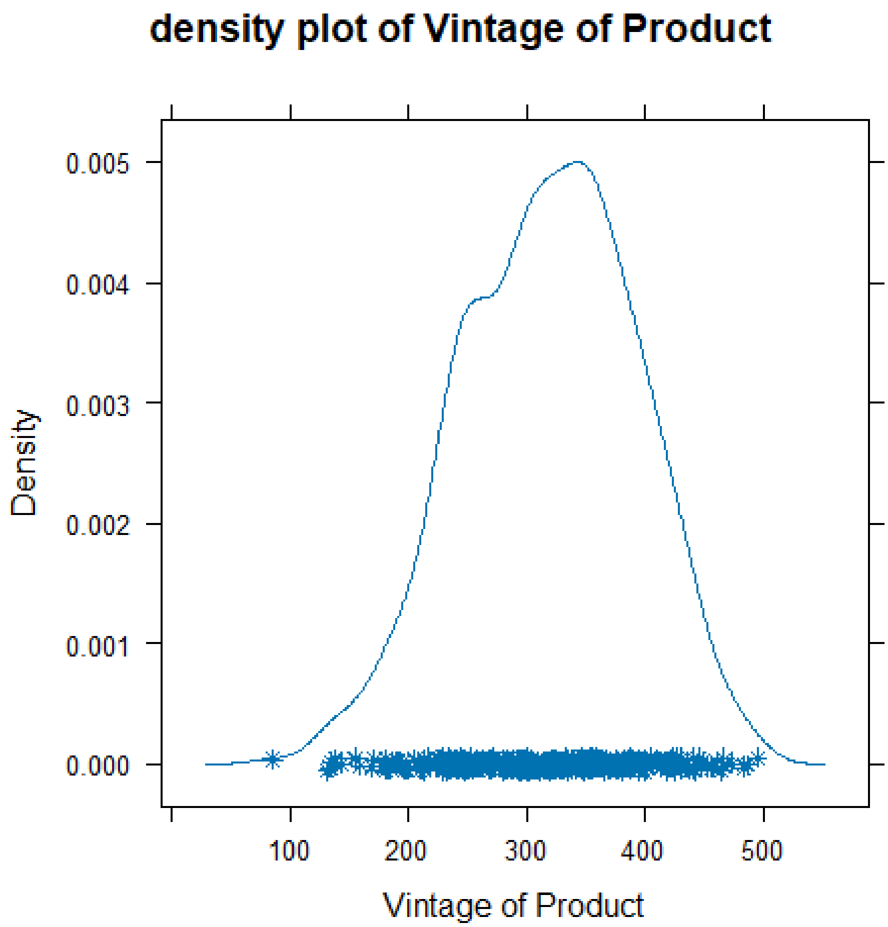

```
boxplot(data_DVH$WT_DVH, horizontal=TRUE, pch=4,col=5, border = 2, main="Box
plot of Weight of shipment")
```



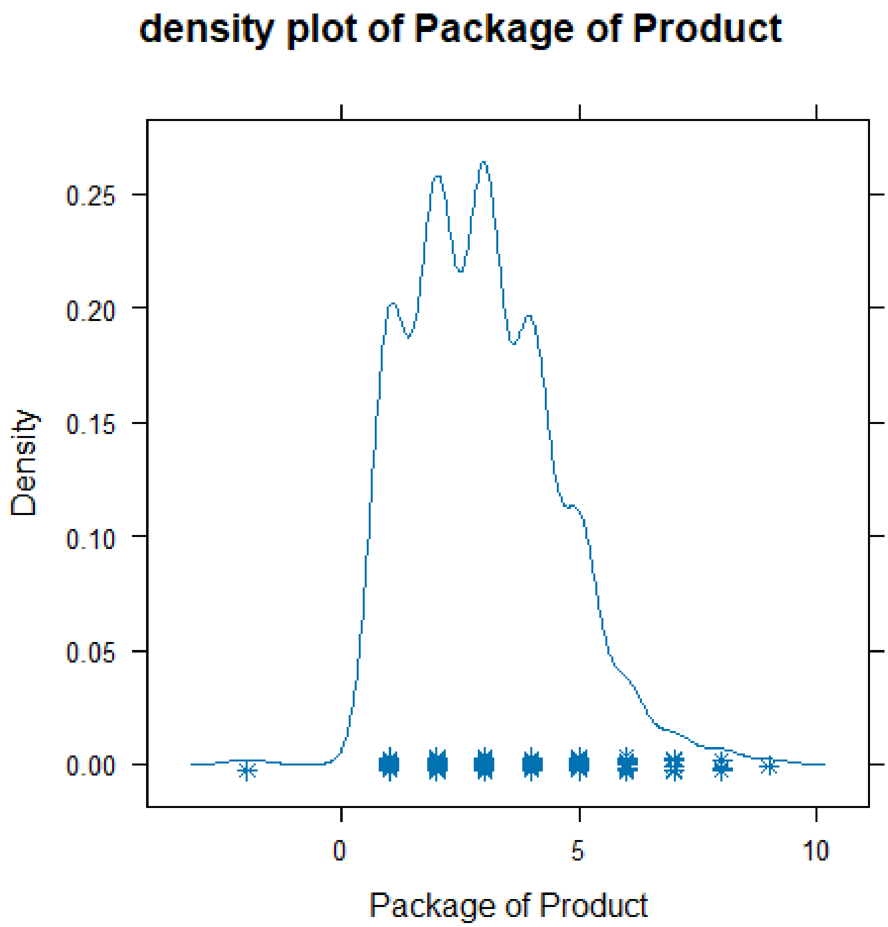

```
densityplot( ~ data_DVH$DL_DVH, pch=8,main="density plot of Time for
Delivery",xlab="Time for Delivery")
```



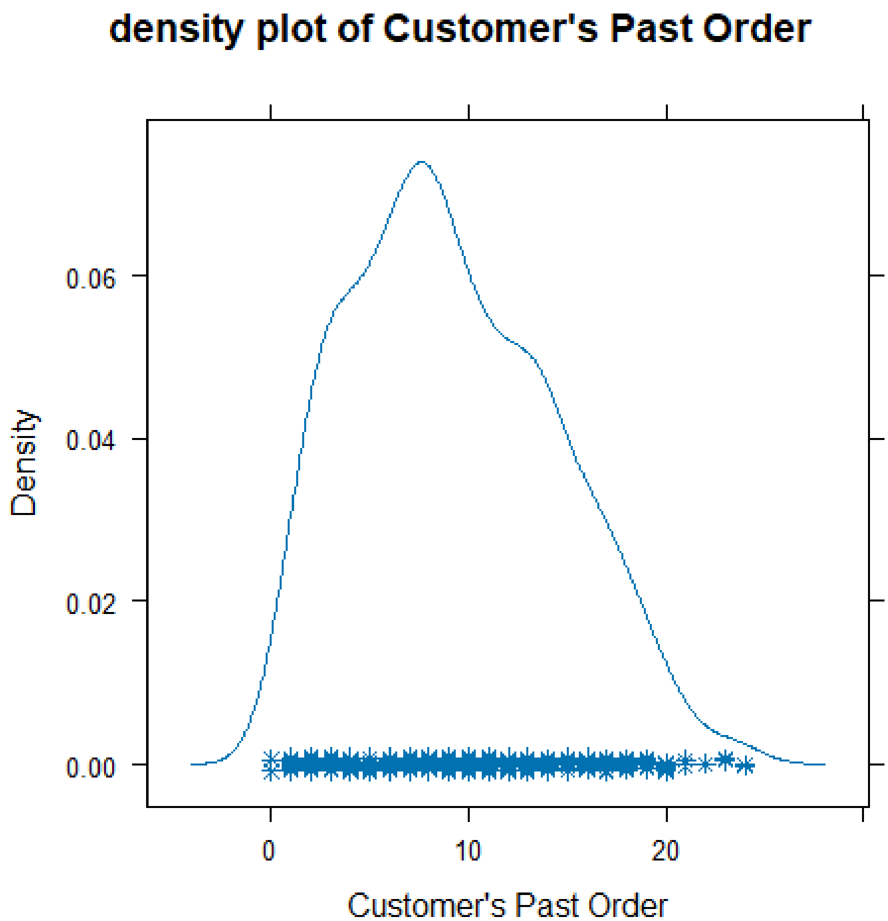

```
densityplot( ~ data_DVH$VN_DVH, pch=8,main="density plot of Vintage of
Product",xlab="Vintage of Product")
```



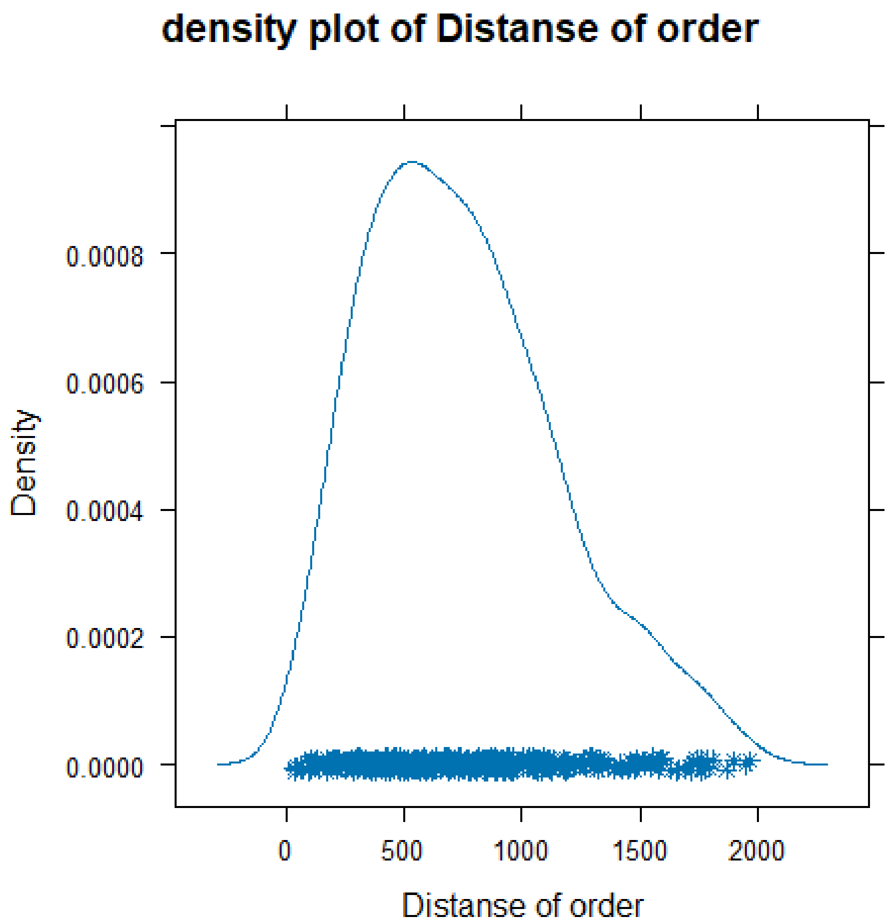

```
densityplot( ~ data_DVH$PG_DVH, pch=8,main="density plot of Package of
Product",xlab="Package of Product")
```



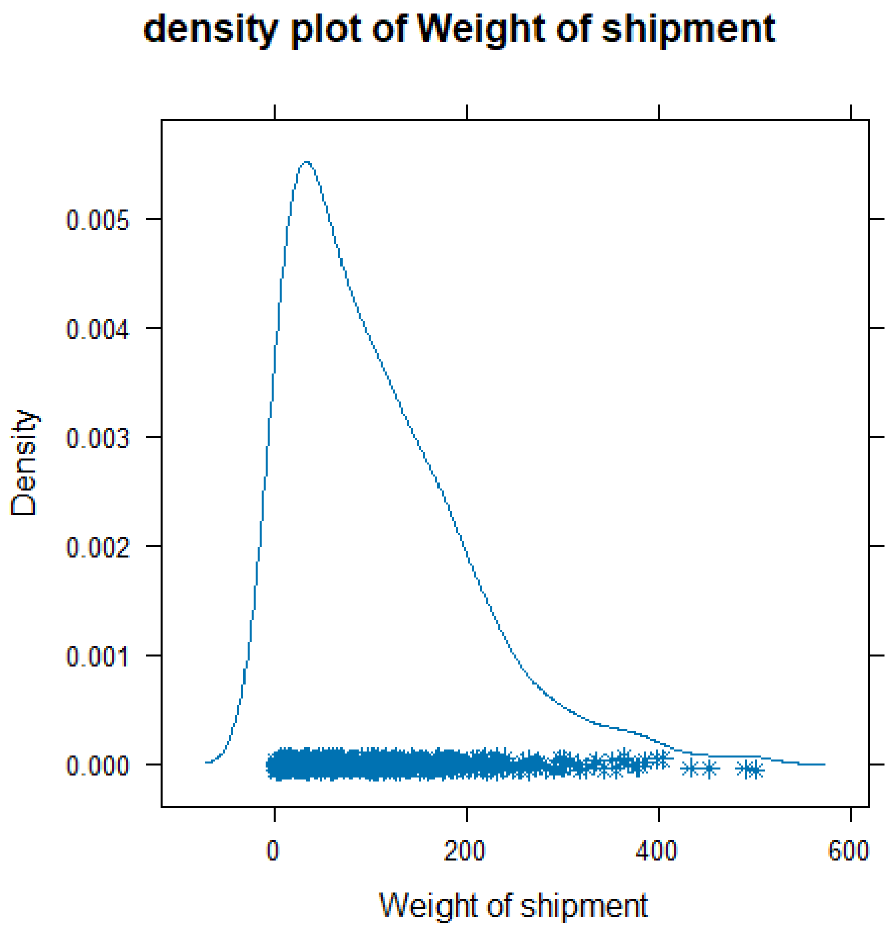

```
densityplot( ~ data_DVH$CS_DVH, pch=8,main="density plot of Customer's Past
Order",xlab="Customer's Past Order")
```




```
densityplot( ~ data_DVH$ML_DVH, pch=8,main="density plot of Distanse of
order",xlab="Distanse of order")
```

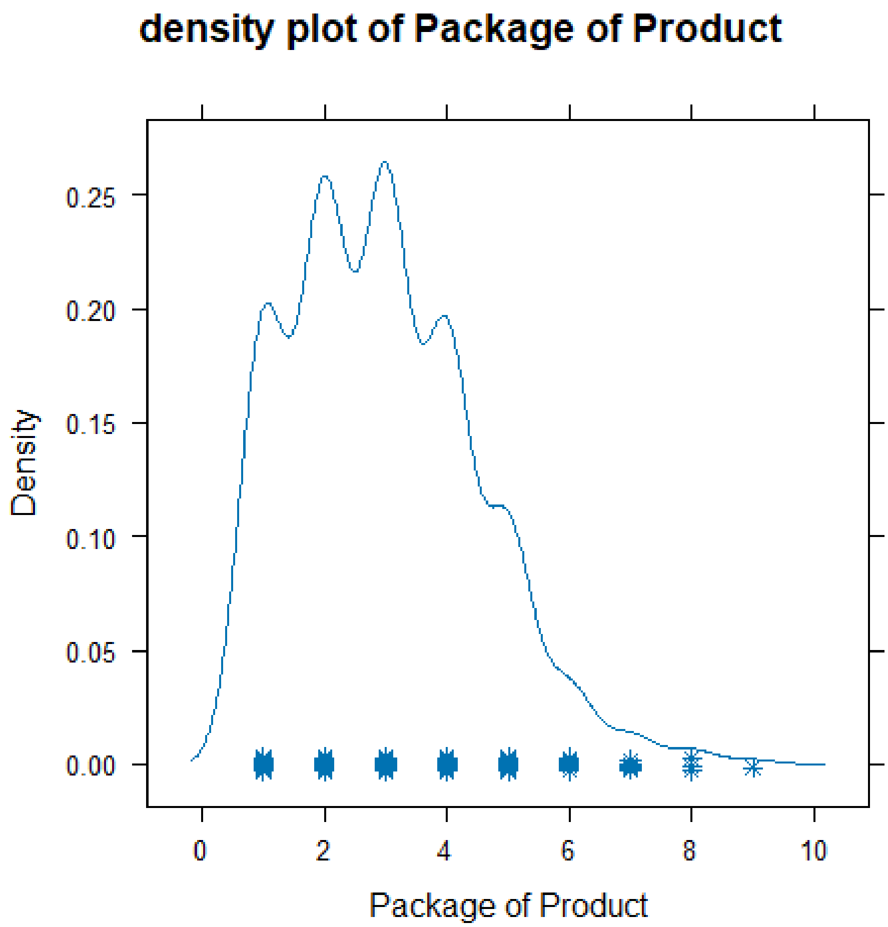



```
densityplot( ~ data_DVH$WT_DVH, pch=8,main="density plot of Weight of
shipment",xlab="Weight of shipment")
```



Interpretation
There are a few outliers presented in some column. But, by observing all outliers, no significant outliers are shown in any column. Since, there is one outlier present in Package of Product, which is below zero value and it will add no value for analysis. hence, i will remove it.

```
nr <- which(data_DVH$PG_DVH < 0)  #Find row number with PG_DVH <= 0
data_DVH <- data_DVH[-c(nr),]

densityplot( ~ data_DVH$PG_DVH, pch=8,main="density plot of Package of
Product",xlab="Package of Product")
```



Interpretation
As you can see from above density plot, outlier is successfully removed from Package of Product column.

3. Using an appropriate technique from class, determine if there is any evidence if one Carrier has faster delivery times than the other. Make sure you explain the approach you took and your conclusions.

```
delivery_time_dp_DVH <- data_DVH$DL_DVH[data_DVH$CR_DVH == "Def Post"]
delivery_time_sd_DVH <- data_DVH$DL_DVH[data_DVH$CR_DVH == "Sup Del"]

t.test(delivery_time_dp_DVH, delivery_time_sd_DVH, var.equal = TRUE)

##
##  Two Sample t-test
##
## data:  delivery_time_dp_DVH and delivery_time_sd_DVH
## t = -6.9147, df = 484, p-value = 0.00000000001488
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -1.3544364 -0.7550164
## sample estimates:
```

```
## mean of x mean of y
##  7.845274  8.900000
```

Interpretation

- The p-value is below 0.05 of t-test . This means that the probability of observing a difference in mean delivery times as extreme or more extreme than the one observed, suggesting strong evidence against the null hypothesis.
- The 95 percent confidence interval for the difference in means is (-1.3544364, -0.7550164). This means that we are 95 percent confident that the true difference in mean delivery times between Def-Post and Sup-Del. Since the interval does not include 0, there is a statistically significant difference in mean delivery times between the two carriers.
- Overall, based on this output, i can conclude that there is evidence to suggest that one carrier has faster delivery times than the other for orders designated as Hazardous.

4. As demonstrated in class, split the dataframe into a training and a test file. This should be a 80/20 split. For the set.seed(), use the last four digits of your student number. The training set will be used to build the following models and the test set will be used to validate them.

```
# Set the seed using the last four digits of your student number
set.seed(6337)

# Split the dataframe into a training and test set using an 80/20 split
train_index_DVH <- sample(1:nrow(data_DVH), floor(0.8*nrow(data_DVH)),
replace = FALSE)
train_data_DVH <- data_DVH[train_index_DVH, ]
test_data_DVH <- data_DVH[-train_index_DVH, ]
```

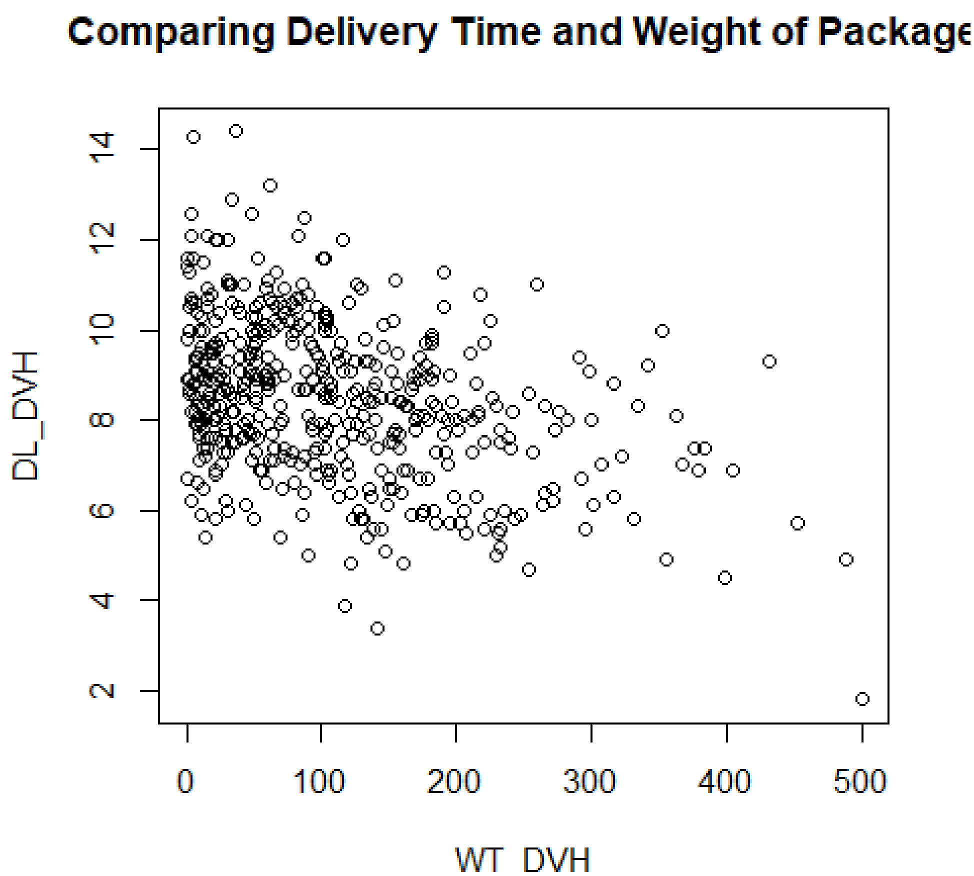
Interpretation
The function set.seed(6337) is used to set a fixed seed value for the random number generator, which ensures that the random split is reproducible.
The sample() function is then used to randomly sample 80% of the rows of data_DVH for the training set.
The floor() function is used to round down to the nearest integer value, as the sample() function requires the sample size to be an integer.
The replace = FALSE argument specifies that sampling should be done without replacement.

2. Simple Linear Regression

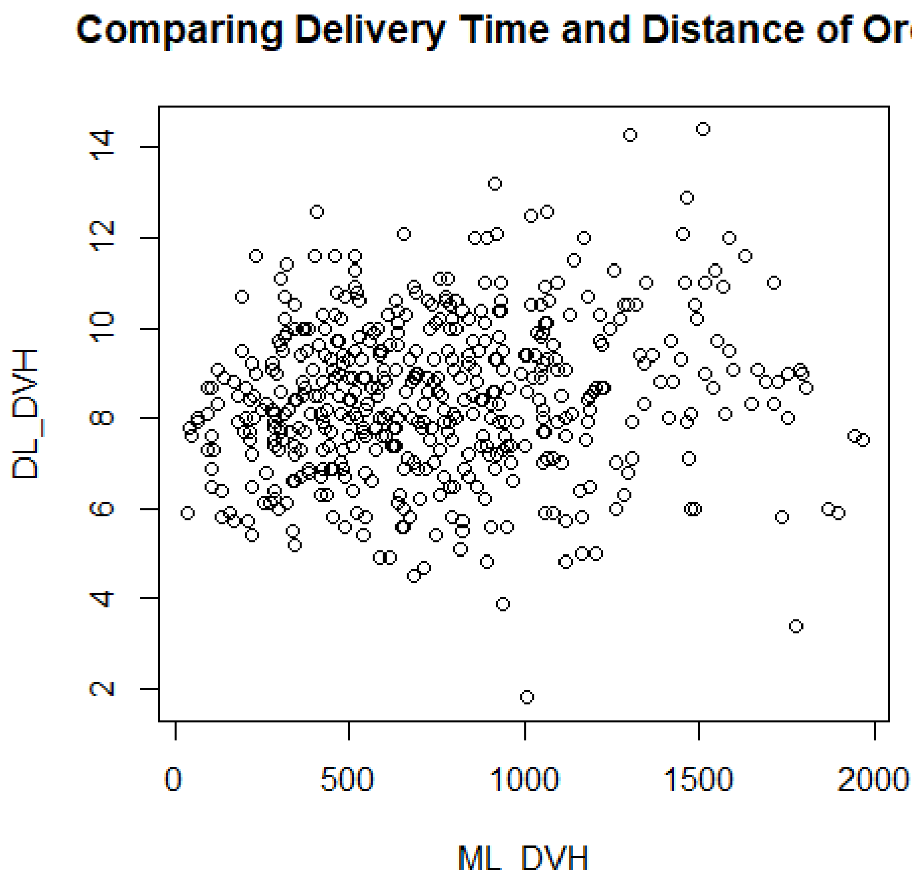
1. Correlations: Create both numeric and graphical correlations (as demonstrated in class) and comment on noteworthy correlations you observe. Are these surprising? Do they make sense?

Here i am representing a relationships between each Numeric individual variable and the target variable, DL.

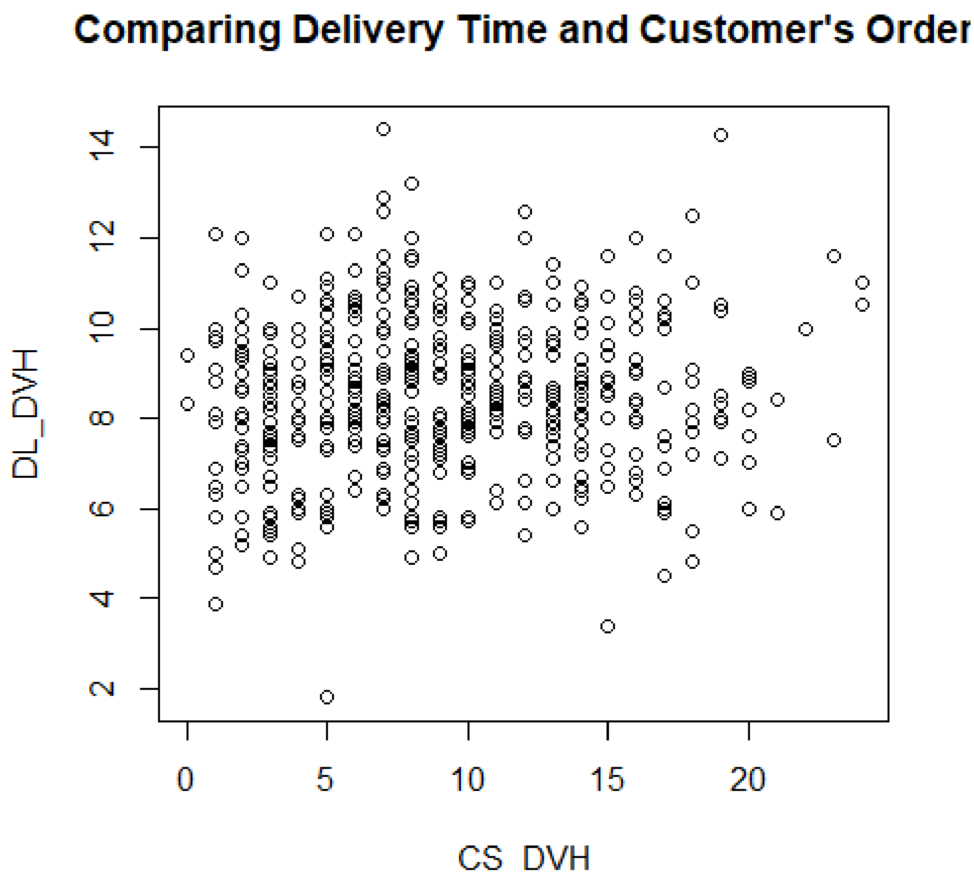
```
# Delivery Time vs Weight (WT)
cor(data_DVH$DL_DVH, data_DVH$WT_DVH, method="spearman")

## [1] -0.3318756

plot(DL_DVH ~ WT_DVH, data = data_DVH,main="Comparing Delivery Time and
Weight of Package")
```



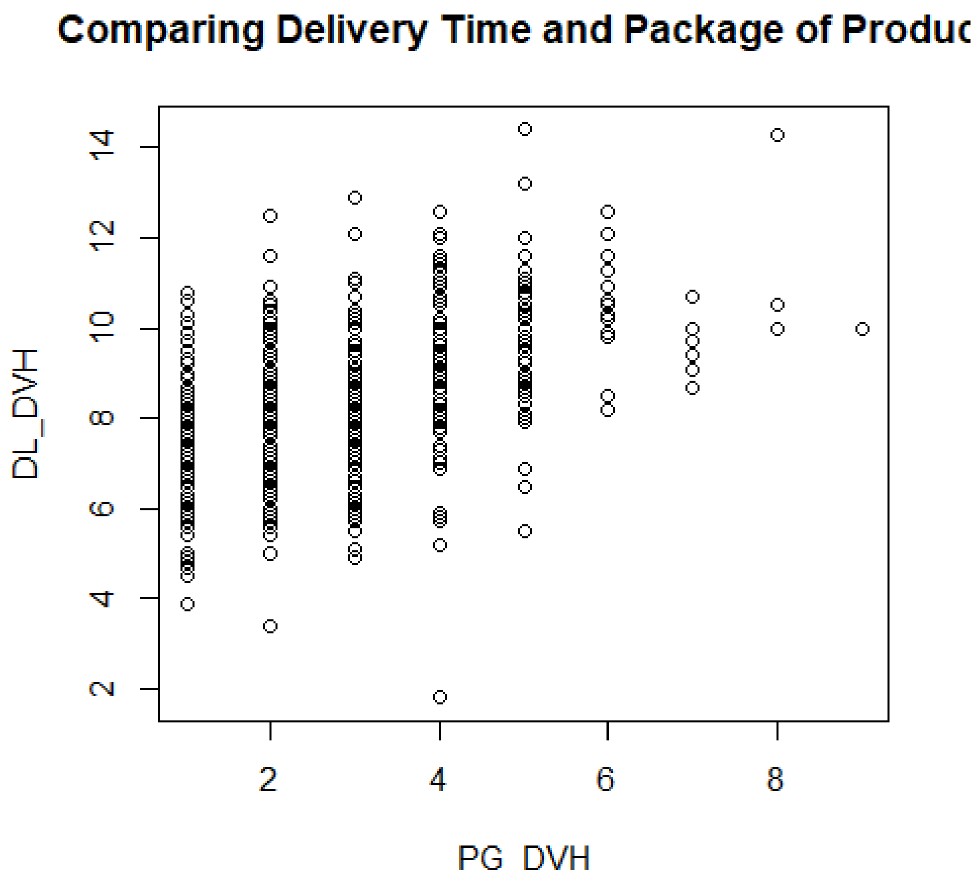

```
# Delivery Time vs Distance the order needs to be delivered (ML)
cor(data_DVH$DL_DVH, data_DVH$ML_DVH, method="spearman")

## [1] 0.1522359

plot(DL_DVH ~ ML_DVH, data = data_DVH,main="Comparing Delivery Time and
Distance of Order")
```



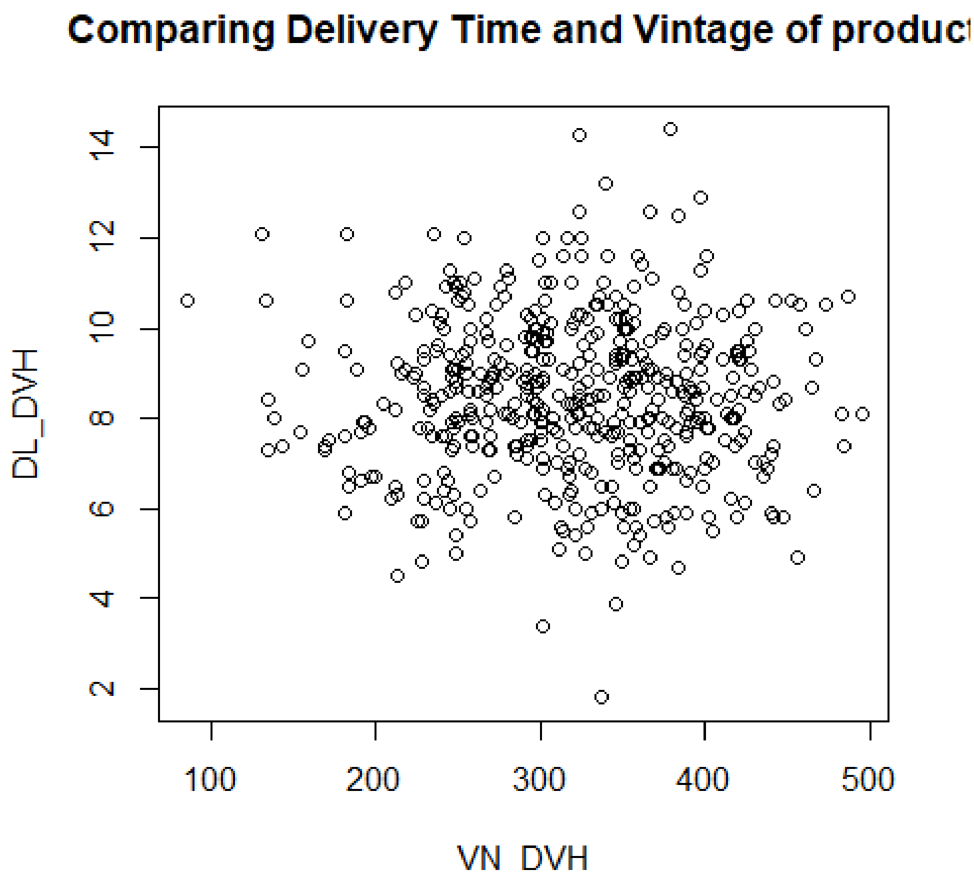

```
# Delivery Time vs orders the customer has made in the past (CS)
cor(data_DVH$DL_DVH, data_DVH$CS_DVH, method="spearman")

## [1] 0.08925397

plot(DL_DVH ~ CS_DVH, data = data_DVH,main="Comparing Delivery Time and
Customer's Order")
```



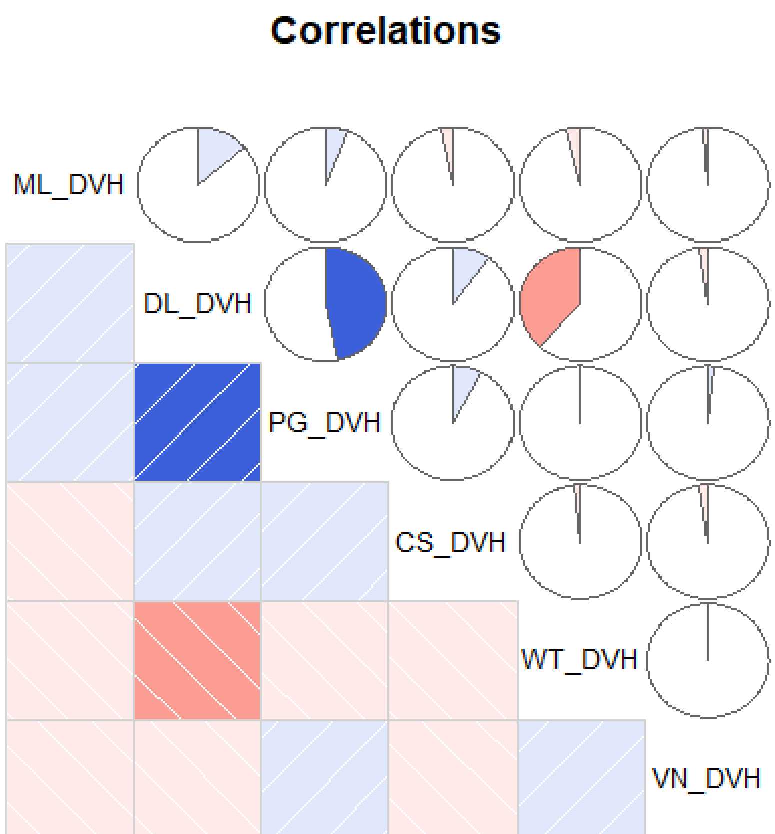

```
# Delivery Time vs Package of Product (PG)
cor(data_DVH$DL_DVH, data_DVH$PG_DVH, method="spearman")

## [1] 0.4635112

plot(DL_DVH ~ PG_DVH, data = data_DVH,main="Comparing Delivery Time and
Package of Product")
```




```r
# Delivery Time vs Vintage of product (VN)
cor(data_DVH$DL_DVH, data_DVH$VN_DVH, method="spearman")

## [1] -0.02708424

plot(DL_DVH ~ VN_DVH, data = data_DVH,main="Comparing Delivery Time and
Vintage of product")
```




```
# Graphical Representation of Correlation
corrgram(data_DVH, order=TRUE, lower.panel=panel.shade,
         upper.panel=panel.pie, text.panel=panel.txt,
         main="Correlations")
```



Interpretation
ML, CS and PG have the Positive linear relationships with DL, with correlation coefficients of 0.15, 0.089, and 0.46, respectively. This indicates that there no relationship of this three variable with DL.
WT and VT have the Negative linear relationships with DL, with correlation coefficients of -0.33 and -0.02726 respectively. Hence, there is no relationship this two variables with DL.

Also, there is a graphical view of correlation presented above. I have used 'data_DVH' variable that has every columns. Here three columns are ignored due to data type is character, which are DM, HZ, CR.

Overall, these correlations and scatter plots are not particularly surprising. By looking at correlation value of ML, CS, PG, WT and VT with DL, it does not make any sense with Delivery Time.

NOTE: *I also have used train_data (train_data_DVH) to find correlation and to check whether there is any difference in result or not. Please follow below code.*

```
# Delivery Time vs Weight (WT)
cor(train_data_DVH$DL_DVH, train_data_DVH$WT_DVH, method="spearman")
```

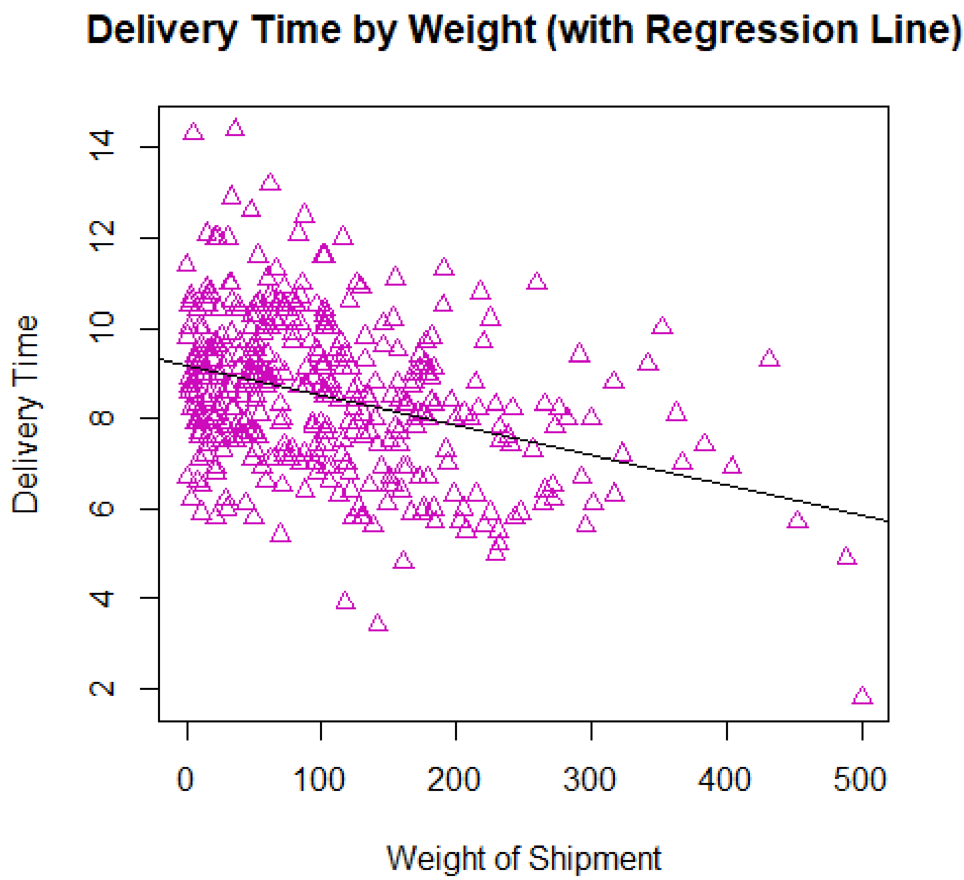
```
## [1] -0.2999729

# Delivery Time vs Distance the order needs to be delivered (ML)
cor(train_data_DVH$DL_DVH, train_data_DVH$ML_DVH, method="spearman")

## [1] 0.145862

# Delivery Time vs orders the customer has made in the past (CS)
cor(train_data_DVH$DL_DVH, train_data_DVH$CS_DVH, method="spearman")

## [1] 0.07734405

# Delivery Time vs Package of Product (PG)
cor(train_data_DVH$DL_DVH, train_data_DVH$PG_DVH, method="spearman")

## [1] 0.4649711

# Delivery Time vs Vintage of product (VN)
cor(train_data_DVH$DL_DVH, train_data_DVH$VN_DVH, method="spearman")

## [1] -0.02158336
```

Interpretation
As you can see from above result, there is no major difference even when checking correlation of train data.

NOTE: *From now, I am using train_data (train_data_DVH) to find simple linear regression. I will check trained data result with test data (test_data_DVH).*

2. Create a simple linear regression model using time for delivery as the dependent variable and weight of the shipment as the independent. Create a scatter plot of the two variables and overlay the regression line.

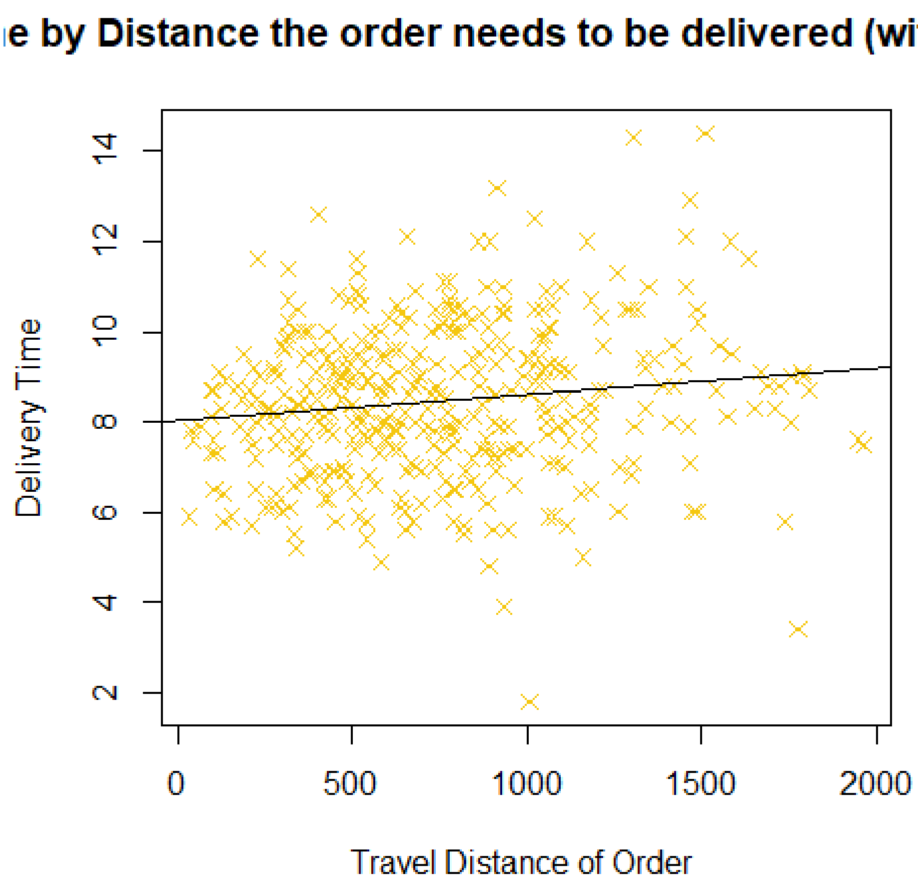
```
WTmodel_DVH <- lm(DL_DVH ~ WT_DVH, data=train_data_DVH)
WTmodel_DVH

##
## Call:
## lm(formula = DL_DVH ~ WT_DVH, data = train_data_DVH)
##
## Coefficients:
## (Intercept)       WT_DVH
##    9.178042    -0.006663

plot(DL_DVH ~ WT_DVH, data=train_data_DVH,
     main="Delivery Time by Weight (with Regression Line)",
     xlab="Weight of Shipment",
     ylab="Delivery Time",
     col=6,
     pch=2)
abline(WTmodel_DVH)
```



3. Create a simple linear regression model using time for delivery as the dependent variable and distance the shipment needs to travel as the independent. Create a scatter plot of the two variables and overlay the regression line.

```
MLmodel_DVH <- lm(DL_DVH ~ ML_DVH, data=train_data_DVH)
MLmodel_DVH

##
## Call:
## lm(formula = DL_DVH ~ ML_DVH, data = train_data_DVH)
##
## Coefficients:
## (Intercept)       ML_DVH
##   8.0436173    0.0005812

plot(DL_DVH ~ ML_DVH, data=train_data_DVH,
     main="Delivery Time by Distance the order needs to be delivered (with
Regression Line)",
     xlab="Travel Distance of Order ",
     ylab="Delivery Time",
     col=7,
```

```
      pch=4)
abline(MLmodel_DVH)
```



4. As demonstrated in class, compare the models (F-Stat, R^2, RMSE for train and test, etc.) Which model is superior? Why?

```
## RMSE of Delivery vs Weight for train data
pred_DlWt_DVH <- predict(WTmodel_DVH, newdata=train_data_DVH)
RMSE_train_DlWt_DVH <- sqrt(mean((train_data_DVH$DL_DVH - pred_DlWt_DVH)^2))

## RMSE of Delivery vs Weight for Test data
pred_DlWt_DVH <- predict(WTmodel_DVH, newdata=test_data_DVH)
RMSE_test_DlWt_DVH <- sqrt(mean((test_data_DVH$DL_DVH - pred_DlWt_DVH)^2))


## RMSE of Delivery vs Distance of Order for Train data
pred_DlMl_DVH <- predict(MLmodel_DVH, newdata=train_data_DVH)
RMSE_train_DlMl_DVH <- sqrt(mean((train_data_DVH$DL_DVH - pred_DlMl_DVH)^2))

## RMSE of Delivery vs Distance of Order for Test data
```

```r
pred_DlMl_DVH <- predict(MLmodel_DVH, newdata=test_data_DVH)
RMSE_test_DlMl_DVH <- sqrt(mean((test_data_DVH$DL_DVH - pred_DlMl_DVH)^2))

cat("----------------------------Model: DL vs WT-----------------------------
")

## ----------------------------Model: DL vs WT-----------------------------

# Summary model of Delivery Time vs Weight of the shipment
summary(WTmodel_DVH)

##
## Call:
## lm(formula = DL_DVH ~ WT_DVH, data = train_data_DVH)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.8319 -1.1850  0.0317  1.0031  5.4618
##
## Coefficients:
##                 Estimate Std. Error t value          Pr(>|t|)
## (Intercept)  9.1780421  0.1242477  73.869           < 2e-16 ***
## WT_DVH      -0.0066627  0.0008988  -7.413 0.000000000000788 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.604 on 386 degrees of freedom
## Multiple R-squared:  0.1246, Adjusted R-squared:  0.1224
## F-statistic: 54.95 on 1 and 386 DF,  p-value: 0.0000000000007884

print(paste0("RMSE Train: ",round(RMSE_train_DlWt_DVH,3)))

## [1] "RMSE Train: 1.6"

print(paste0("RMSE Test: ",round(RMSE_test_DlWt_DVH,3)))

## [1] "RMSE Test: 1.611"

cat("----------------------------Model: DL vs ML-----------------------------
")

## ----------------------------Model: DL vs ML-----------------------------

# Summary model of Delivery Time vs Distance the order needs to be delivered
summary(MLmodel_DVH)

##
## Call:
## lm(formula = DL_DVH ~ ML_DVH, data = train_data_DVH)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
```

```
## -6.8295 -1.0824 -0.0176  1.1206  5.4973
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 8.0436173  0.1809287  44.457  < 2e-16 ***
## ML_DVH      0.0005812  0.0002107   2.759  0.00608 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.698 on 386 degrees of freedom
## Multiple R-squared:  0.01934,    Adjusted R-squared:  0.0168
## F-statistic: 7.611 on 1 and 386 DF,  p-value: 0.006077
```

```
print(paste0("RMSE Train: ",round(RMSE_train_DlMl_DVH,3)))
```

```
## [1] "RMSE Train: 1.694"
```

```
print(paste0("RMSE Test: ",round(RMSE_test_DlMl_DVH,3)))
```

```
## [1] "RMSE Test: 1.804"
```

Interpretation

- P-value of F-statistic: The p-value of the F-statistic for the DL vs WT model is below 0.05, which indicates that the model is significant at a high level of confidence. The p-value of the F-statistic for the DL vs ML model is above 0.05, which indicates that the model is marginally significant at a 5% level of significance.

- Adjusted R-squared value: The adjusted R-squared value for the DL vs WT model is 0.1224, which means that the model explains 12.24% of the variability in the response variable, after adjusting for the number of predictors. The adjusted R-squared value for the DL vs ML model is 0.0168, which means that the model explains only 1.68% of the variability in the response variable, after adjusting for the number of predictors. Therefore, the DL vs WT model has a higher adjusted R-squared value and is a better fit for the data.

- Coefficient and t-test value: For the DL vs WT model, the coefficient estimate for WT_DVH is -0.0066627 with a t-value of -7.413 . This means that the weight of the sample has a significant negative effect on the DL_DVH. For the DL vs ML model, the coefficient estimate for ML_DVH is 0.0005812 with a t-value of 2.759. This means that the distance of order has a marginally significant positive effect on the DL_DVH.

- RMSE value: The root mean squared error (RMSE) is a measure of how well the model fits the data. A lower RMSE value indicates a better fit of the model to the data. Therefore, the DL vs WT model has a lower RMSE value for both train (1.6) and test(1.611) data compared to DL vs ML model.

Based on the above analysis, I can conclude that the DL vs WT model is a better fit for the data as compared to the DL vs ML model. The DL vs WT model has a lower p-value of F-statistic, higher adjusted R-squared value, significant negative effect of predictor variable, and lower RMSE value for both train and test data.

3. Model Development – Multivariate

1. create two models, one using all the variables and the other using backward selection. This should be built using the train set created in Step 2. For each model interpret and comment on the main measures we discussed in class (including RMSE for train and test).

NOTE: *As dataset has three different variables (DM, HZ, CR) with Character data type. And, This task required to use all variables for model training. Hence, I am changing this data type into Factor type here.*

```
train_data_DVH <- as.data.frame(unclass(train_data_DVH), stringsAsFactors =
TRUE)
head(train_data_DVH,3)

##   DL_DVH VN_DVH PG_DVH CS_DVH ML_DVH DM_DVH HZ_DVH   CR_DVH WT_DVH
## 1    7.7    300      1     10    201      C      N Def Post     50
## 2    1.8    337      4      5   1008      C      N Def Post    500
## 3    6.8    331      2     10    547      C      N Def Post     22

test_data_DVH <- as.data.frame(unclass(test_data_DVH), stringsAsFactors =
TRUE)
head(test_data_DVH,3)

##   DL_DVH VN_DVH PG_DVH CS_DVH ML_DVH DM_DVH HZ_DVH   CR_DVH WT_DVH
## 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    5.4    321      1      2    221      C      N Def Post     14
```

Interpretation

DM_DVH (char) –> DM_DVH (fctr)
HZ_DVH (char) –> HZ_DVH (fctr)
CR_DVH (char) –> CR_DVH (fctr)

```
full_model_DVH = lm(DL_DVH ~ . ,
            data=train_data_DVH, na.action=na.omit)

back_model_DVH = step(full_model_DVH, direction="backward", details=TRUE)

## Start:  AIC=188.38
## DL_DVH ~ VN_DVH + PG_DVH + CS_DVH + ML_DVH + DM_DVH + HZ_DVH +
##     CR_DVH + WT_DVH
##
##          Df Sum of Sq    RSS    AIC
## - VN_DVH  1     0.163 602.08 186.48
## - CS_DVH  1     1.254 603.17 187.18
## <none>                601.92 188.38
## - ML_DVH  1     5.351 607.27 189.81
```

```
## - DM_DVH  1    14.764 616.68 195.78
## - HZ_DVH  1    23.845 625.76 201.45
## - CR_DVH  1    95.922 697.84 243.75
## - WT_DVH  1   112.924 714.84 253.09
## - PG_DVH  1   253.046 854.96 322.54
##
## Step:  AIC=186.48
## DL_DVH ~ PG_DVH + CS_DVH + ML_DVH + DM_DVH + HZ_DVH + CR_DVH +
##     WT_DVH
##
##          Df Sum of Sq    RSS    AIC
## - CS_DVH  1     1.278 603.36 185.31
## <none>                602.08 186.48
## - ML_DVH  1     5.364 607.45 187.92
## - DM_DVH  1    14.632 616.71 193.80
## - HZ_DVH  1    23.912 625.99 199.59
## - CR_DVH  1    96.706 698.79 242.28
## - WT_DVH  1   113.138 715.22 251.29
## - PG_DVH  1   252.904 854.99 320.55
##
## Step:  AIC=185.31
## DL_DVH ~ PG_DVH + ML_DVH + DM_DVH + HZ_DVH + CR_DVH + WT_DVH
##
##          Df Sum of Sq    RSS    AIC
## <none>                603.36 185.31
## - ML_DVH  1     5.566 608.93 186.87
## - DM_DVH  1    14.940 618.30 192.80
## - HZ_DVH  1    24.344 627.70 198.65
## - CR_DVH  1    97.350 700.71 241.34
## - WT_DVH  1   113.386 716.75 250.12
## - PG_DVH  1   256.092 859.45 320.57
```

Interpretation

Two models are created above with Full model and Backword selection process. This is build using training set (train_data_DVH) from original data.

```
# RMSE on Train data
pred_train_DVH <- predict(full_model_DVH, newdata=train_data_DVH)
RMSE_train_full_DVH <- sqrt(mean((train_data_DVH$DL_DVH - pred_train_DVH)^2))

# RMSE on Test data
pred_test_DVH <- predict(full_model_DVH, newdata=test_data_DVH)
RMSE_test_full_DVH <- sqrt(mean((test_data_DVH$DL_DVH - pred_test_DVH)^2))



# RMSE on Train data
pred_train_DVH <- predict(back_model_DVH, newdata=train_data_DVH)
RMSE_train_back_DVH <- sqrt(mean((train_data_DVH$DL_DVH - pred_train_DVH)^2))
```

```r
# RMSE on Test data
pred_test_DVH <- predict(back_model_DVH, newdata=test_data_DVH)
RMSE_test_back_DVH <- sqrt(mean((test_data_DVH$DL_DVH - pred_test_DVH)^2))

cat("-----------------------------------Full Model-------------------------------------")

## ----------------------------------Full Model------------------------------------

summary(full_model_DVH)

##
## Call:
## lm(formula = DL_DVH ~ ., data = train_data_DVH, na.action = na.omit)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.9039 -0.7394  0.0046  0.7548  4.0732
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)    7.2203129  0.3934589  18.351  < 2e-16 ***
## VN_DVH        -0.0002743  0.0008555  -0.321 0.748698
## PG_DVH         0.5353189  0.0424094  12.623  < 2e-16 ***
## CS_DVH         0.0111471  0.0125456   0.889 0.374821
## ML_DVH         0.0002898  0.0001579   1.836 0.067203 .
## DM_DVHI        0.4290673  0.1407274   3.049 0.002458 **
## HZ_DVHN       -0.7607633  0.1963375  -3.875 0.000126 ***
## CR_DVHSup Del  1.0285258  0.1323441   7.772 7.36e-14 ***
## WT_DVH        -0.0060170  0.0007136  -8.432 7.19e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.26 on 379 degrees of freedom
## Multiple R-squared:  0.4696, Adjusted R-squared:  0.4584
## F-statistic: 41.95 on 8 and 379 DF,  p-value: < 2.2e-16

print(paste0("RMSE Train: ",round(RMSE_train_full_DVH,3)))

## [1] "RMSE Train: 1.246"

print(paste0("RMSE Test: ",round(RMSE_test_full_DVH,3)))

## [1] "RMSE Test: 1.195"

cat("-----------------------------------Backward Selection Model-----------------------------------")

## ----------------------------------Backward Selection Model-----------------------------------
```

```
summary(back_model_DVH)

##
## Call:
## lm(formula = DL_DVH ~ PG_DVH + ML_DVH + DM_DVH + HZ_DVH + CR_DVH +
##     WT_DVH, data = train_data_DVH, na.action = na.omit)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.8398 -0.7124  0.0053  0.7675  4.1894
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)    7.2304997  0.2645361  27.333  < 2e-16 ***
## PG_DVH         0.5371867  0.0422428  12.717  < 2e-16 ***
## ML_DVH         0.0002954  0.0001575   1.875 0.061579 .
## DM_DVHI        0.4306375  0.1402053   3.071 0.002283 **
## HZ_DVHN       -0.7680991  0.1959042  -3.921 0.000105 ***
## CR_DVHSup Del  1.0340624  0.1318878   7.840 4.55e-14 ***
## WT_DVH        -0.0060278  0.0007124  -8.462 5.73e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.258 on 381 degrees of freedom
## Multiple R-squared:  0.4683, Adjusted R-squared:   0.46
## F-statistic: 55.94 on 6 and 381 DF,  p-value: < 2.2e-16

print(paste0("RMSE Train: ",round(RMSE_train_back_DVH,2)))

## [1] "RMSE Train: 1.25"

print(paste0("RMSE Test: ",round(RMSE_test_back_DVH,2)))

## [1] "RMSE Test: 1.2"
```

Interpretation

- The first model (Full Model) has 8 predictor variables and an adjusted R-squared of 0.4584, indicating that about 45.84% of the variability in the response variable is explained by the predictor variables. The F-statistic is significant at $p < 2.2e-16$, indicating that at least one of the predictor variables is significantly related to the response variable. The RMSE for the training set is 1.246 and for the test set is 1.195.

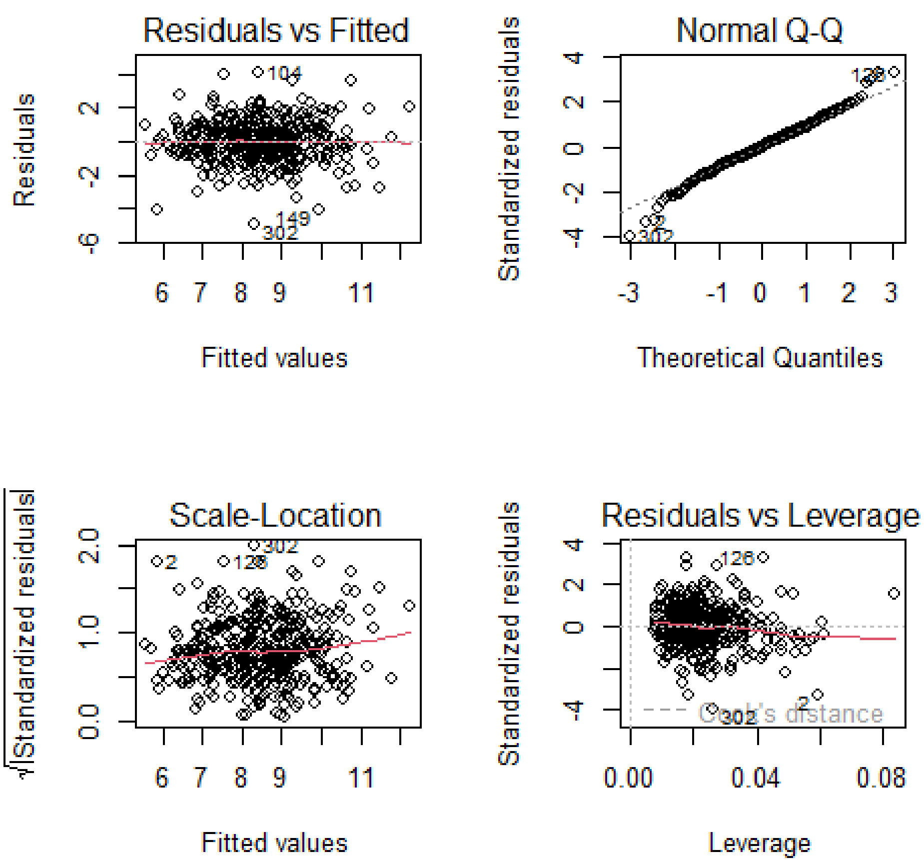
- The second model (Backward Selection Model) has 6 predictor variables and an adjusted R-squared of 0.46, indicating that about 46.0% of the variability in the response variable is explained by the predictor variables. The F-statistic is significant at $p < 2.2e-16$, indicating that at least one of the predictor variables is significantly related to the response variable. The RMSE for the training set is 1.25 and for the test set is 1.2.

Comparing the two models, i see that the second model has a slightly higher adjusted R-squared and slightly lower RMSE for the training set, but a slightly higher RMSE for the test set. Additionally, the second model has fewer predictor variables, which could be desirable for simplicity.

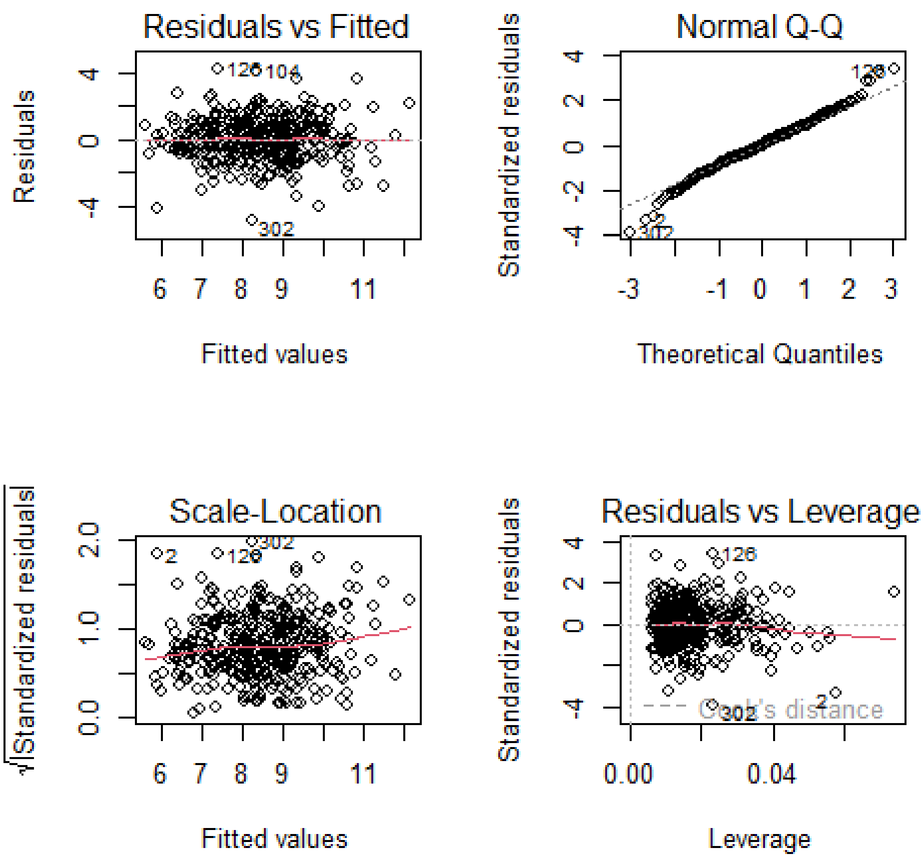
However, in terms of the predictor variables that are included, and it is possible that different predictor variables may be more or less important depending on the specific context of the analysis. Therefore, the best model would ultimately depend on a variety of factors and should be selected based on the specific needs of the analysis.

4. Model Evaluation – Verifying Assumptions - Multivariate

```
# Full Model
par(mfrow = c(2, 2))
plot(full_model_DVH)
```




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

# Backword Model
par(mfrow = c(2, 2))
plot(back_model_DVH)
```




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

Interpretation

Error terms mean of zero: In both models, the residuals vs. fitted values plots show no clear pattern, indicating that the error terms have a mean of zero.

Constant variance: In the full model, the residuals vs. fitted values plot shows some curvature, indicating that the variance of the error terms may not be constant. In the backward model, the residuals vs. fitted values plot shows a more constant spread of residuals, suggesting that the constant variance assumption may be met.

Normally distributed: In both models, the normal Q-Q plots show some deviation from normality, particularly in the tails, suggesting that the normality assumption may not be fully met.

```
## Creating Model and Residual vectors ##

full_res_DVH <- residuals(full_model_DVH)
back_res_DVH <- residuals(back_model_DVH)
```

```
#Check Normality Numericaly
shapiro.test(full_res_DVH)

##
##  Shapiro-Wilk normality test
##
## data:  full_res_DVH
## W = 0.98951, p-value = 0.007046

shapiro.test(back_res_DVH)

##
##  Shapiro-Wilk normality test
##
## data:  back_res_DVH
## W = 0.9896, p-value = 0.007453
```

Interpretation

The Shapiro-Wilk test is commonly used to examine if the error terms in a linear regression model are normally distributed.
For both models, the p-values obtained from the test are greater than the usual significance level of 0.05, which means that there is no enough evidence to reject the null hypothesis. Therefore, the normality assumption of the error terms is met for both models.

5. Final Recommendation - Multivariate

Answer: The backward selection model is a better choice for making predictions on new data. Because, it has a simpler structure, with only six predictor variables (5/6 passed in t-test), making it easier to interpret and potentially more robust. Additionally, It has higher Adjusted R squared value and lower RMSE value compared to Full model.