Mahavir Education Trust's

Shah & Anchor Kutchhi Engineering College,

Chembur, Mumbai 400 088

UG Program in Information Technology

Name: Sayyam Gada

Class-Batch: BE5-A

Roll No: 02

Experiment No.: 10					
Date of Performance:	**15/04/2022**				
Date of Submission:	**22/04/2022**				
Program formation/ Execution/ ethical practices (07)	**Documentation (02)**	**Timely Submission (03)**	**Viva Answer (03)**	**Experiment Marks (15)**	**Teacher Signature with date**
6	**2**	**3**	**3**	**14**	

Experiment No. 10

Aim: R for data Science (Mini Project)

- Extracting data from a large Dataset
- Exploratory Analysis
- Use Mining Algorithm

Visualizations & interpretation of results

Lab Outcome(s):

2.ITL804.6: Apply the knowledge of R gained to data Analytics for real life applications.

RPL Project on E-Commerce Shipping

By

Name of the Students	Class	Roll No.
Sayyam Gada	BE-5	02
Jugal Gala	BE-5	04
Nikita Bhilare	BE-5	20

Guide

Ms. Manya Gidwani

Department of Information Technology

Shah & Anchor Kutchhi Engineering College, Mumbai

2021-2022

Approval for RPL Mini Project Report for Fourth Year semester VIII

R Programming Lab

This RPL mini-project report entitled "E-Commerce Shipping" by Sayyam Gada, Jugal Gala, Nikita Bhilare is approved in partial fulfilment of the requirement for the R Programming Lab of Final year Engineering.

Examiners

1.____________________________

2.____________________________

Guide

1.____________________________

2.____________________________

1. Problem Statement

An international electronics e-commerce company wants to discover key insights from their customer database. A key performance indicator for this company is timely delivery. It can be modeled as a function of parameters like mode of shipment, cost of the product, number of prior purchases made by the customer, etc. Hence, in this project, we tried to devise a prediction model for the same using random forest on the data set. It'll predict timely delivery. We've used R's graphical tools to build some random visualizations on the mentioned data set where we've studied various parameters that help in determining the timely delivery.

2. Exploratory Analysis

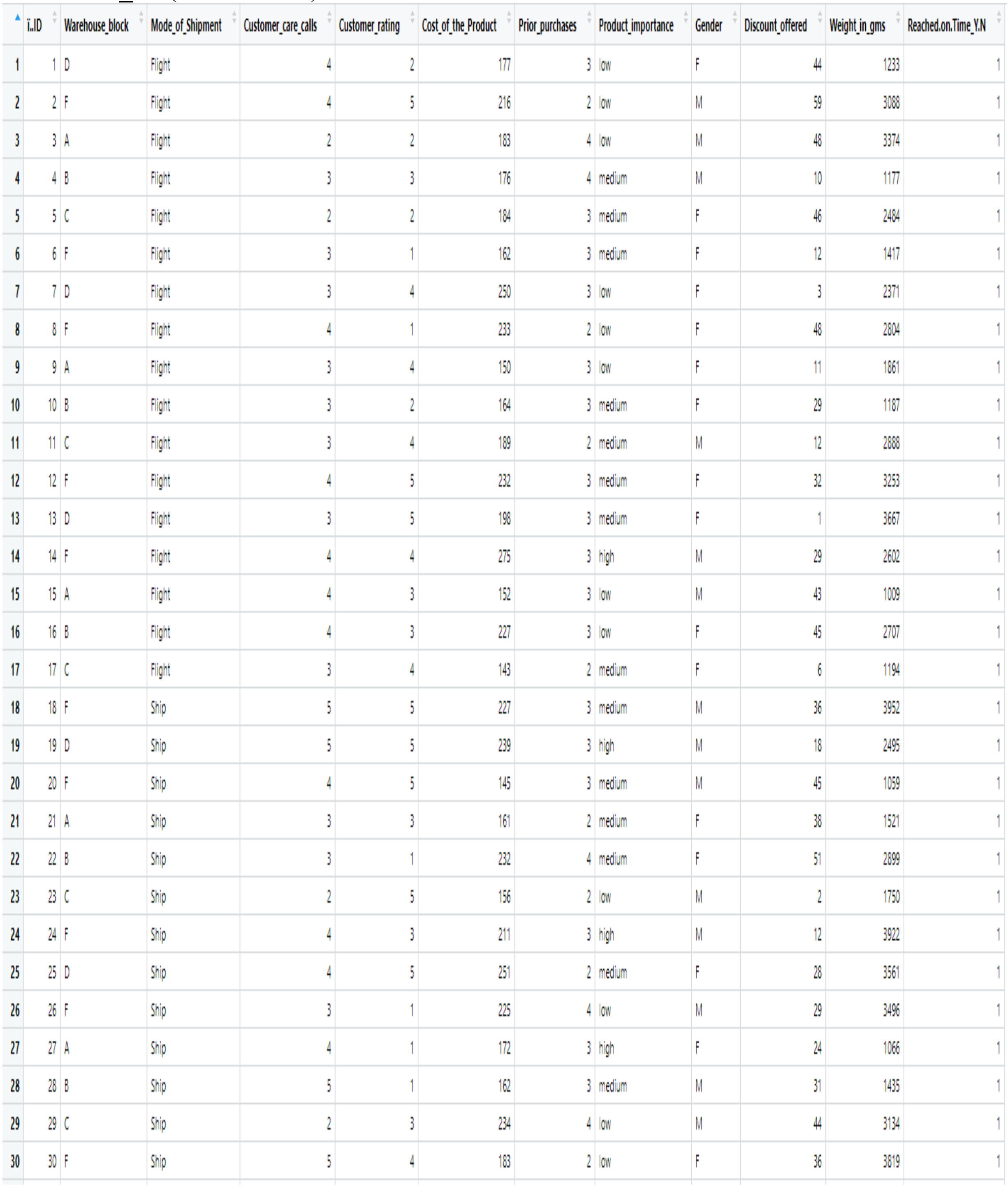
No.	Attribute	Type	Label	Distinct Val	Unique Val	Min Val	Max Val	Count
1	id	numeric		10999	10999	1	10999	10999
2	warehouse block	nominal	A	5				1833
			B					1833
			C					1833
			D					1844
			F					3668
3	mode of shippment	nominal	Flight	3				1777
			Ship					7462
			Road					1760
4	customer care cells	numeric		6	0	2	7	
5	customer rating	numeric		5	0	1	5	10999
6	cost of product	numeric		215	10999	96	310	10999
7	prior purchases	numeric		8	10999	2	10	10999
8	product importance	nominal	Low	3	10999			5297
			Medium					4754
			high					948
9	gender	nominal	F					5545
			M					5454
10	discount offered	numeric		65	0	1	65	10999
11	weight in gms	numeric		4034	1160	1001	7846	10999
12	reached on time YN	numeric		2	0	0	1	10999

3. Data Preprocessing

Data Loading:

Download the dataset from Kaggle and save it .csv extension. Load the dataset. Create a subset of data-name as ‘data’ which contains the essential attributes on which analysis is to be done.

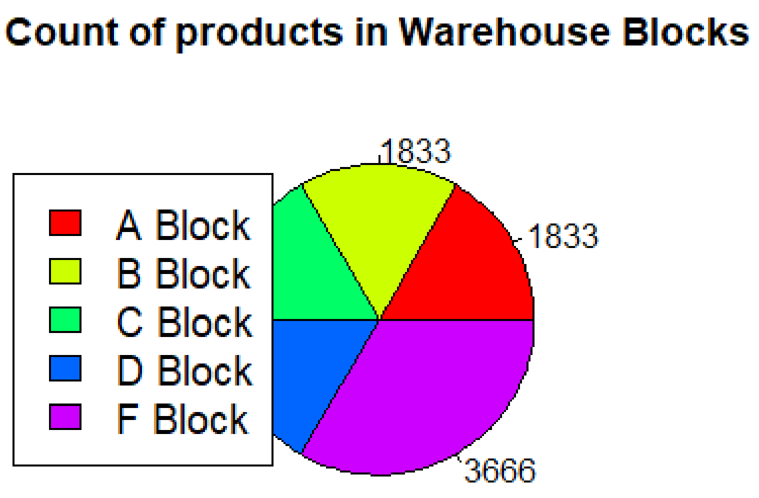
```
data = read_csv("./Train.csv")
```



	i..ID	Warehouse_block	Mode_of_Shipment	Customer_care_calls	Customer_rating	Cost_of_the_Product	Prior_purchases	Product_importance	Gender	Discount_offered	Weight_in_gms	Reached.on.Time_Y.N
1	1	D	Flight	4	2	177	3	low	F	44	1233	1
2	2	F	Flight	4	5	216	2	low	M	59	3088	1
3	3	A	Flight	2	2	183	4	low	M	48	3374	1
4	4	B	Flight	3	3	176	4	medium	M	10	1177	1
5	5	C	Flight	2	2	184	3	medium	F	46	2484	1
6	6	F	Flight	3	1	162	3	medium	F	12	1417	1
7	7	D	Flight	3	4	250	3	low	F	3	2371	1
8	8	F	Flight	4	1	233	2	low	F	48	2804	1
9	9	A	Flight	3	4	150	3	low	F	11	1861	1
10	10	B	Flight	3	2	164	3	medium	F	29	1187	1
11	11	C	Flight	3	4	189	2	medium	M	12	2888	1
12	12	F	Flight	4	5	232	3	medium	F	32	3253	1
13	13	D	Flight	3	5	198	3	medium	F	1	3667	1
14	14	F	Flight	4	4	275	3	high	M	29	2602	1
15	15	A	Flight	4	3	152	3	low	M	43	1009	1
16	16	B	Flight	4	3	227	3	low	F	45	2707	1
17	17	C	Flight	3	4	143	2	medium	F	6	1194	1
18	18	F	Ship	5	5	227	3	medium	M	36	3952	1
19	19	D	Ship	5	5	239	3	high	M	18	2495	1
20	20	F	Ship	4	5	145	3	medium	M	45	1059	1
21	21	A	Ship	3	3	161	2	medium	F	38	1521	1
22	22	B	Ship	3	1	232	4	medium	F	51	2899	1
23	23	C	Ship	2	5	156	2	low	M	2	1750	1
24	24	F	Ship	4	3	211	3	high	M	12	3922	1
25	25	D	Ship	4	5	251	2	medium	F	28	3561	1
26	26	F	Ship	3	1	225	4	low	M	29	3496	1
27	27	A	Ship	4	1	172	3	high	F	24	1066	1
28	28	B	Ship	5	1	162	3	medium	M	31	1435	1
29	29	C	Ship	2	3	234	4	low	M	44	3134	1
30	30	F	Ship	5	4	183	2	low	F	36	3819	1

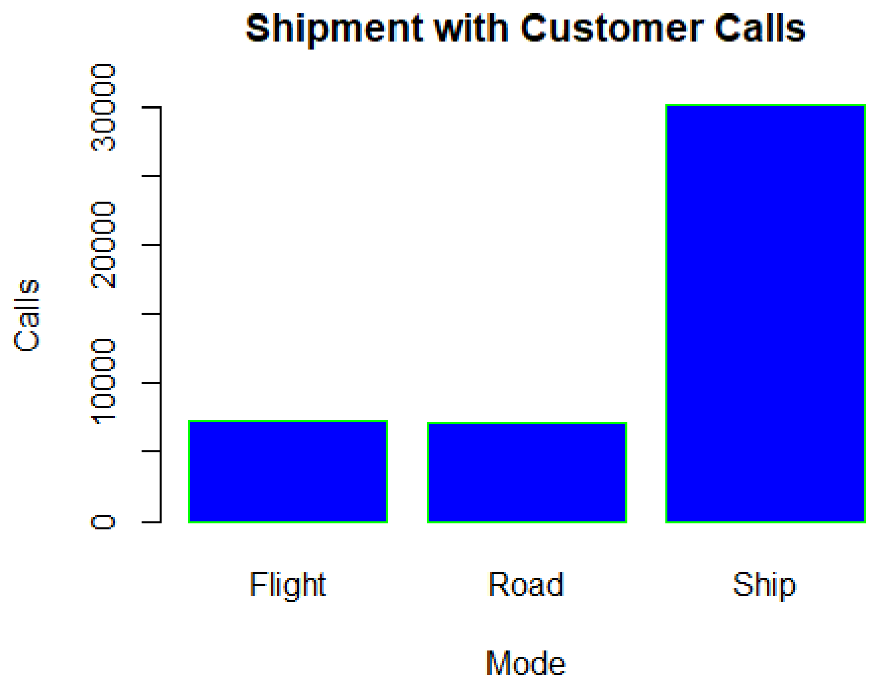
4. Visualization

A) Count of products in warehouse blocks



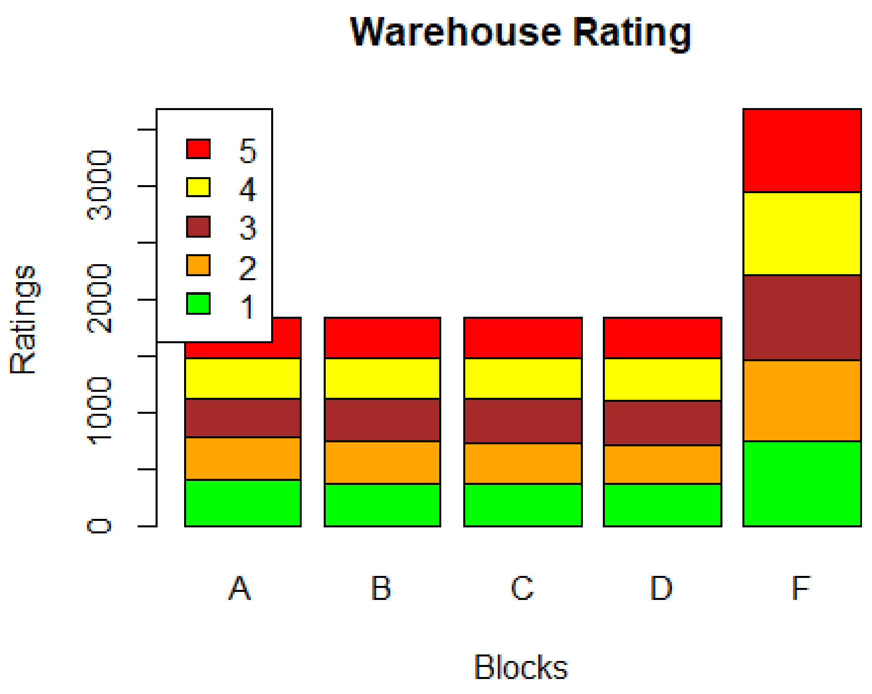
11000 products distributed across 5 Warehouse Blocks.

B) Shipment with customer calls



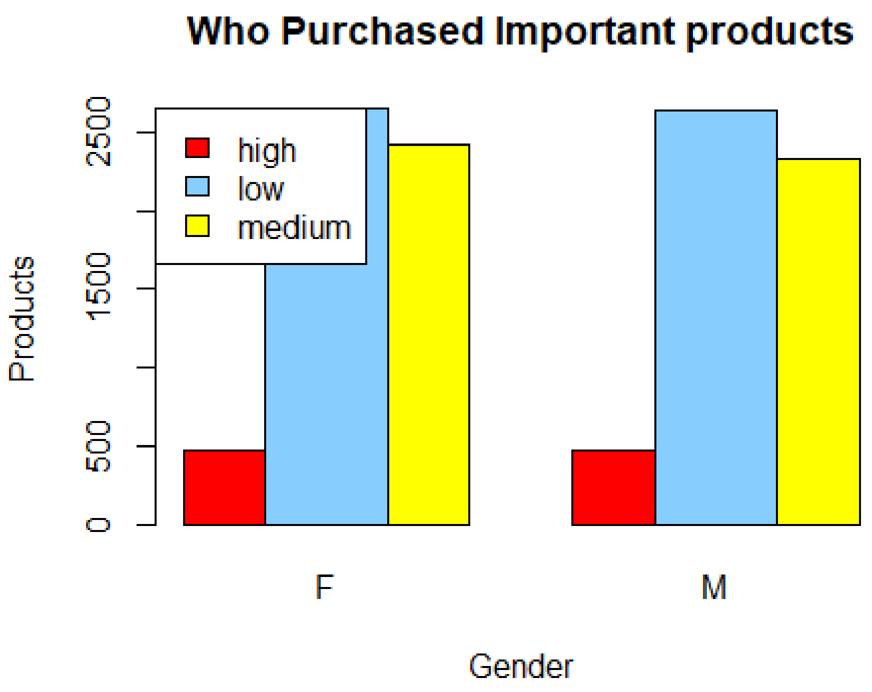
Number of customer calls across each mode of transportation.

C) Warehouse Rating



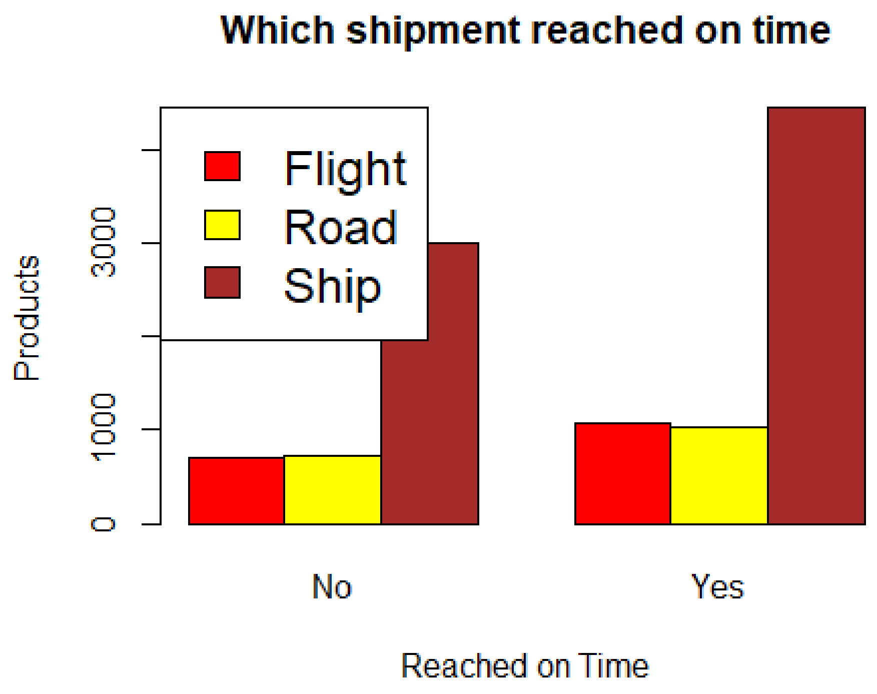
Number of ratings ranging from 1 to 5 for each warehouse.

D) Who purchased important products



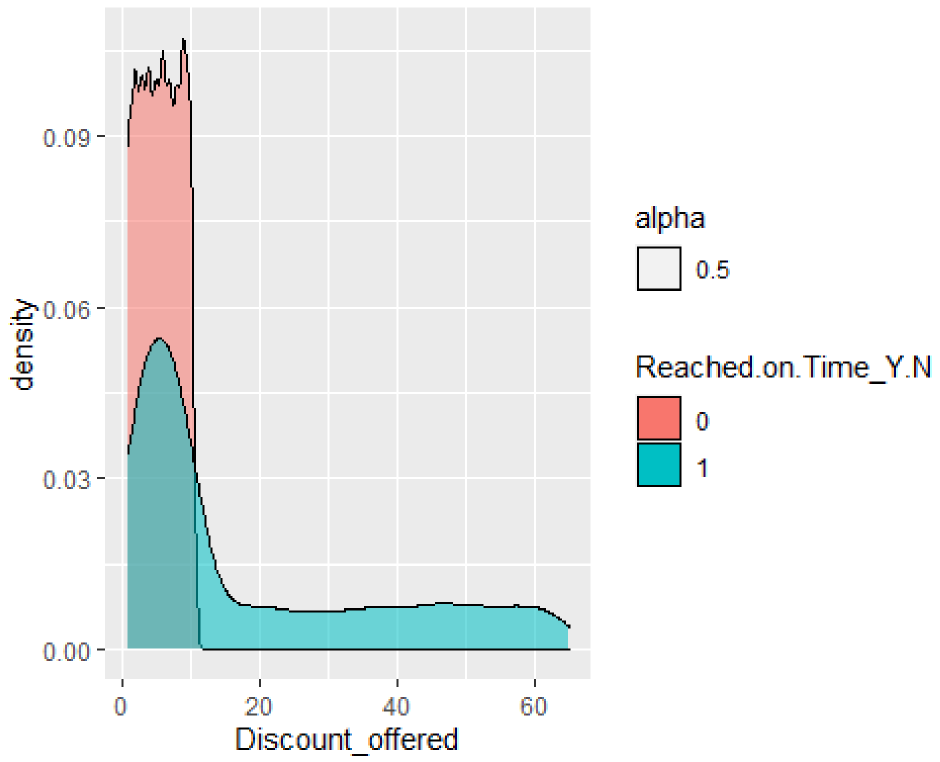
Product purchases based on product importance and gender of customer.

E) Which Shipment reached on time



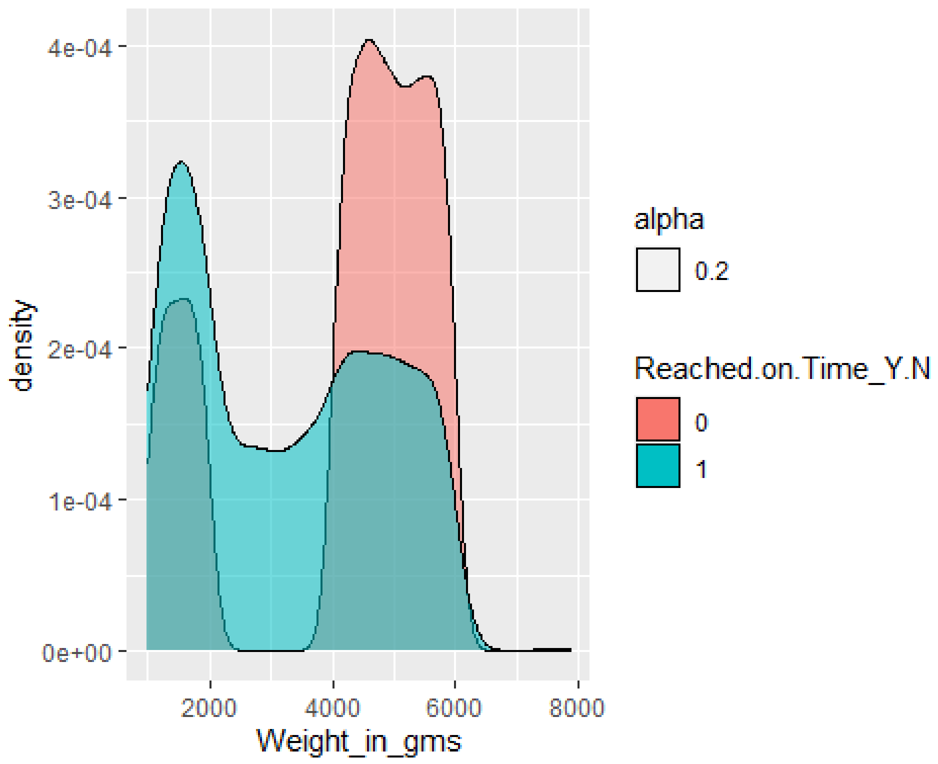
Number of products across each mode of transport based on delivery time.

F) Discount density graph



Products with more than 12% discount will reach on time.

G) Weight density graph



Products with weight less than 3500 grams will have higher chances of being delivered on time.

5. Algorithm

Random Forest is a popular machine learning algorithm that belongs to the supervised learning technique. It can be used for both Classification and Regression problems in ML. It is based on the concept of ensemble learning**,** which is a process of combining multiple classifiers to solve a complex problem and to improve the performance of the model.

As the name suggests, "Random Forest is a classifier that contains a number of decision trees on various subsets of the given data set and takes the average to improve the predictive accuracy of that data set." Instead of relying on one decision tree, the random forest takes the prediction from each tree and based on the majority votes of predictions, and it predicts the final output. The greater number of trees in the forest leads to higher accuracy and prevents the problem of over fitting.

Random Forest Algorithm:

1) Select random K data points from the training set.

2) Build the decision trees associated with the selected data points (Subsets).

3) Choose the number N for decision trees that you want to build.

4) Repeat Step 1 & 2.

5) For new data points, find the predictions of each decision tree, and assign the new data points to the category that wins the majority votes.

Highlights:

1. We divided the data set into 90% training and 10% testing.
2. Area under the curve=0.7002
3. Hence, the model yielded an accuracy of 70.02%

6. Conclusion

Shipping goods across the world will always involve variability in some ways. This is not a deterministic environment. What we sought to accomplish with this project was not to eliminate this variability but rather mitigate its effects on the supply chains it impacts through better prediction of time spent in transit. Today, consumers are keen to maximize their shopping utility by comprehensively considering all possible channels.

Hence, in this project, we devised a prediction model for predicting the shipment time period for a product using random forest algorithm on the data set and determined that discount offered and weight of the product had most impact on shipment delivery time. It predicted timely delivery and yielded an accuracy of 70.02%. We also used R's graphical tools to build some random visualizations on the data set where we studied various parameters that helped in determining the timely delivery. The visualizations include charts like the bar chats, pie chart, and density graphs.

7. Code

```
######## readr provides a fast and friendly way to read rectangular data
#       (like csv, tsv, and fwf). It is designed to flexibly parse many types
#       of data found in the wild, while still cleanly failing when data
#       unexpectedly changes.
library(readr)
#### read_csv() is a special case of the more general read_delim(). It's useful
#    for reading comma separated values (flat file data).
data = read_csv("./Train.csv")

## Rows: 10999 Columns: 12
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (4): Warehouse_block, Mode_of_Shipment, Product_importance, Gender
## dbl (8): ID, Customer_care_calls, Customer_rating, Cost_of_the_Product, Prio...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.

#ggplot lib
library(tidyverse)

## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --
## v ggplot2 3.3.5     v dplyr   1.0.8
## v tibble  3.1.6     v stringr 1.4.0
## v tidyr   1.2.0     v forcats 0.5.1
## v purrr   0.3.4
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()

#confusion matrix lib
library(caret)

## Loading required package: lattice
##
## Attaching package: 'caret'
##
## The following object is masked from 'package:purrr':
##
##     lift

#stats
print(is.data.frame(data))

## [1] TRUE

print(ncol(data))

## [1] 12

print(nrow(data))

## [1] 10999

str(data)

## spec_tbl_df [10,999 x 12] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##  $ ID                 : num [1:10999] 1 2 3 4 5 6 7 8 9 10 ...
##  $ Warehouse_block    : chr [1:10999] "D" "F" "A" "B" ...
##  $ Mode_of_Shipment   : chr [1:10999] "Flight" "Flight" "Flight" "Flight" ...
##  $ Customer_care_calls: num [1:10999] 4 4 2 3 2 3 3 4 3 3 ...
## $ Customer_rating    : num [1:10999] 2 5 2 3 2 1 4 1 4 2 ...
## $ Cost_of_the_Product: num [1:10999] 177 216 183 176 184 162 250 233 150 164 ...
## $ Prior_purchases    : num [1:10999] 3 2 4 4 3 3 3 2 3 3 ...
## $ Product_importance : chr [1:10999] "low" "low" "low" "medium" ...
## $ Gender             : chr [1:10999] "F" "M" "M" "M" ...
## $ Discount_offered   : num [1:10999] 44 59 48 10 46 12 3 48 11 29 ...
## $ Weight_in_gms      : num [1:10999] 1233 3088 3374 1177 2484 ...
## $ Reached.on.Time_Y.N: num [1:10999] 1 1 1 1 1 1 1 1 1 1 ...
## - attr(*, "spec")=
##  .. cols(
##  ..   ID = col_double(),
##  ..   Warehouse_block = col_character(),
##  ..   Mode_of_Shipment = col_character(),
##  ..   Customer_care_calls = col_double(),
##  ..   Customer_rating = col_double(),
##  ..   Cost_of_the_Product = col_double(),
##  ..   Prior_purchases = col_double(),
##  ..   Product_importance = col_character(),
##  ..   Gender = col_character(),
##  ..   Discount_offered = col_double(),
##  ..   Weight_in_gms = col_double(),
##  ..   Reached.on.Time_Y.N = col_double()
##  .. )
## - attr(*, "problems")=<externalptr>
```

```
summary(data)
```

```
##        ID        Warehouse_block    Mode_of_Shipment   Customer_care_calls
## Min.   :    1   Length:10999       Length:10999       Min.   :2.000
## 1st Qu.: 2750   Class :character   Class :character   1st Qu.:3.000
## Median : 5500   Mode  :character   Mode  :character   Median :4.000
## Mean   : 5500                                         Mean   :4.054
## 3rd Qu.: 8250                                         3rd Qu.:5.000
## Max.   :10999                                         Max.   :7.000
## Customer_rating Cost_of_the_Product Prior_purchases  Product_importance
## Min.   :1.000   Min.   : 96.0       Min.   : 2.000   Length:10999
## 1st Qu.:2.000   1st Qu.:169.0       1st Qu.: 3.000   Class :character
## Median :3.000   Median :214.0       Median : 3.000   Mode  :character
## Mean   :2.991   Mean   :210.2       Mean   : 3.568
## 3rd Qu.:4.000   3rd Qu.:251.0       3rd Qu.: 4.000
## Max.   :5.000   Max.   :310.0       Max.   :10.000
##    Gender          Discount_offered Weight_in_gms  Reached.on.Time_Y.N
## Length:10999       Min.   : 1.00    Min.   :1001   Min.   :0.0000
## Class :character   1st Qu.: 4.00    1st Qu.:1840   1st Qu.:0.0000
## Mode  :character   Median : 7.00    Median :4149   Median :1.0000
##                    Mean   :13.37    Mean   :3634   Mean   :0.5967
##                    3rd Qu.:10.00    3rd Qu.:5050   3rd Qu.:1.0000
##                    Max.   :65.00    Max.   :7846   Max.   :1.0000
```

```
#### Set seed for the Random Number Generator for reproducibility.
set.seed(845481568)
#### sample() takes a sample of the specified size from the data, with or without
#   replacement.
index = sample(1:nrow(data), 10000)

## Divide data into training data and testing data, which is further divided
#  into explanatory variables (x) and response variables (y).
X_training = data[index,-ncol(data)]
```

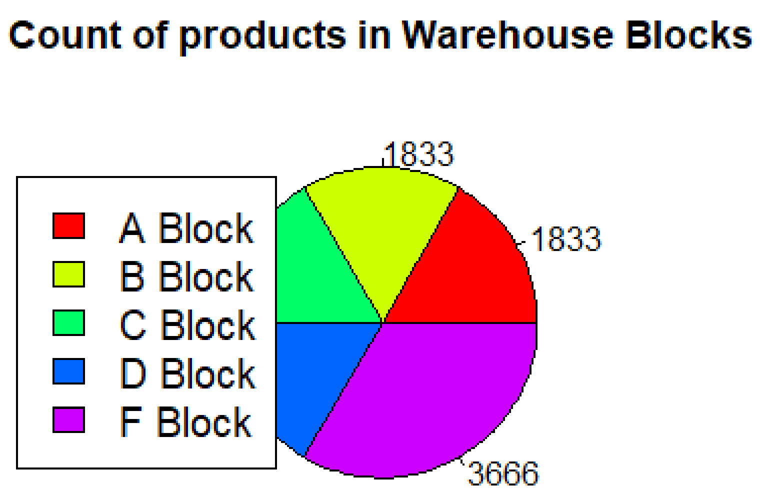
```
y_training = data[index, ncol(data)]
X_testing = data[-index, -ncol(data)]
y_testing = data[-index, ncol(data)]

######## dplyr provides a grammar of data manipulation, providing a consistent
#       set of verbs that solve the most common data manipulation challenges.
library(dplyr)
#### %>% pipes an object forward into a function or call expression.
#    %>% f(y) is equivalent to f(x, y)
#### mutate() adds new variables and preserves existing ones. New variables
#    overwrite existing variables of the same name.
#### across() makes it easy to apply the same transformation to multiple
#    columns, allowing you to use select() semantics inside in "data-masking"
#    functions like summarise() and mutate().
#### where(fn) selects the variables for which the function returns TRUE.
#### is.character returns TRUE or FALSE depending on whether its argument is of
#    character type or not.
#### as.factor coerces its argument to a factor.
X_training <-
  X_training %>% mutate(across(where(is.character), as.factor))
X_testing <-
  X_testing %>% mutate(across(where(is.character), as.factor))
y_training$Reached.on.Time_Y.N <-
  y_training$Reached.on.Time_Y.N %>% as.factor
y_testing$Reached.on.Time_Y.N <-
  y_testing$Reached.on.Time_Y.N %>% as.factor

library(randomForest)
```

```
## randomForest 4.7-1
## Type rfNews() to see new features/changes/bug fixes.
##
## Attaching package: 'randomForest'
##
## The following object is masked from 'package:dplyr':
##
##     combine
##
## The following object is masked from 'package:ggplot2':
##
##     margin
```

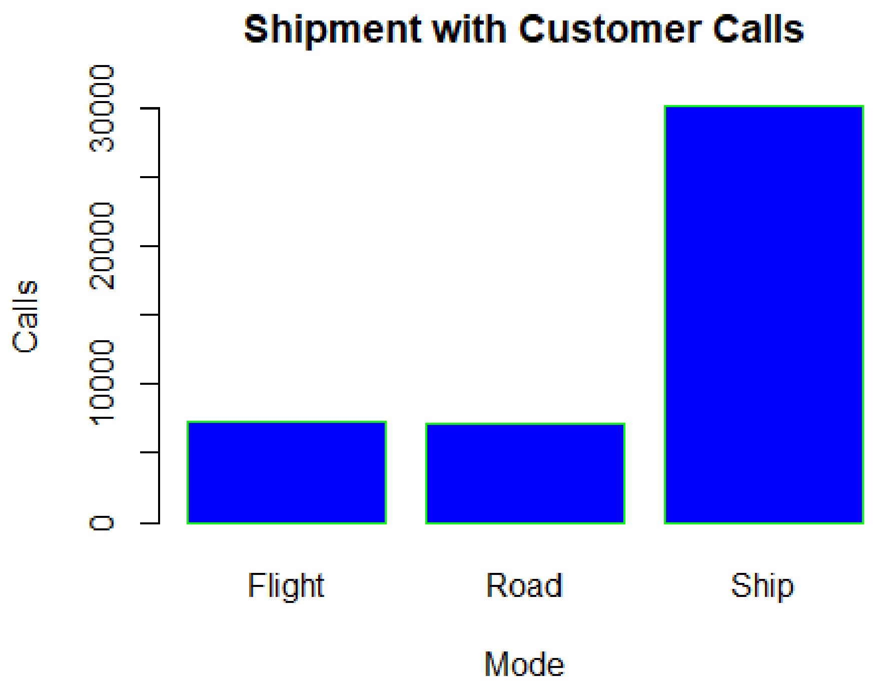
```
model <-
  randomForest(y_training$Reached.on.Time_Y.N ~ . - ID - Gender - Warehouse_block,
               X_training)
prediction <- predict(model, X_testing)

library(pROC)
```

```
## Type 'citation("pROC")' for a citation.
##
## Attaching package: 'pROC'
##
## The following objects are masked from 'package:stats':
##
##     cov, smooth, var
```

```
#Display confusion matrix
confusionMatrix(data = prediction, reference = y_testing$Reached.on.Time_Y.N, positive = "1")

## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 304 244
##          1  78 373
##
##                Accuracy : 0.6777
##                  95% CI : (0.6477, 0.7066)
##     No Information Rate : 0.6176
##     P-Value [Acc > NIR] : 4.515e-05
##
##                   Kappa : 0.3698
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.6045
##             Specificity : 0.7958
##          Pos Pred Value : 0.8271
##          Neg Pred Value : 0.5547
##              Prevalence : 0.6176
##          Detection Rate : 0.3734
##    Detection Prevalence : 0.4515
##       Balanced Accuracy : 0.7002
##
##        'Positive' Class : 1
##

print(auc(y_testing$Reached.on.Time_Y.N, as.numeric(prediction)))

## Setting levels: control = 0, case = 1
## Setting direction: controls < cases

## Area under the curve: 0.7002

#Visualization

#display product count in each warehouse
ware <- data.frame(table(data$Warehouse_block))
pie(ware$Freq,
   labels = ware$Freq,
   main = "Count of products in Warehouse Blocks",
   col = rainbow(length(ware$Var1)))
   legend("left",c("A Block","B Block","C Block", "D Block","F Block"),
        cex = 1.25,
        fill = rainbow(length(ware$Var1)))
```

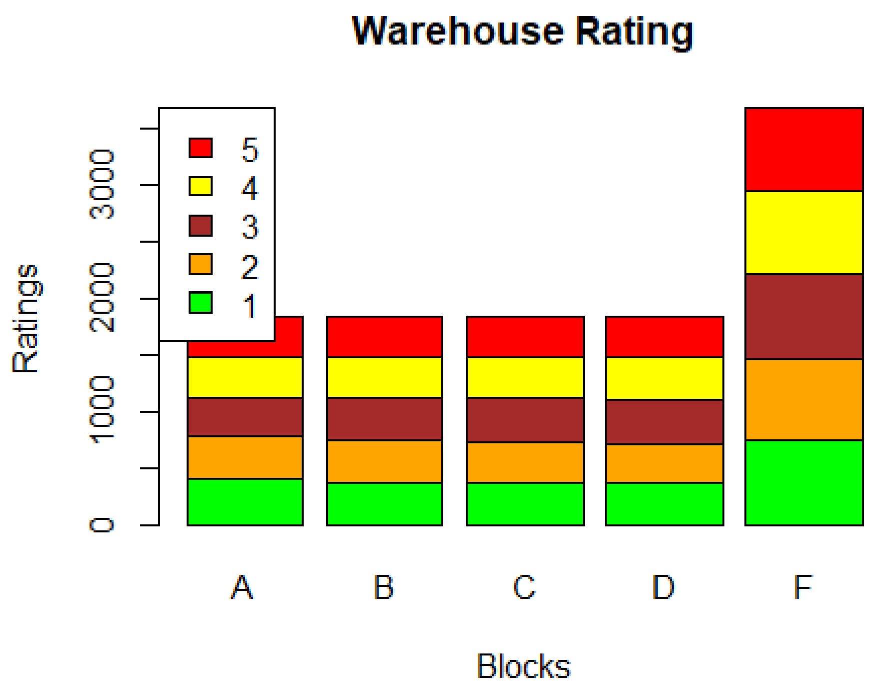


```
#display shipments with customer calls count
ship <- aggregate(data$Customer_care_calls ~ data$Mode_of_Shipment, FUN = sum)
barplot(ship$`data$Customer_care_calls`,names.arg = ship$`data$Mode_of_Shipment`,
     xlab = "Mode", ylab = "Calls", main = "Shipment with Customer Calls",
     col = "blue", border = "green")
```

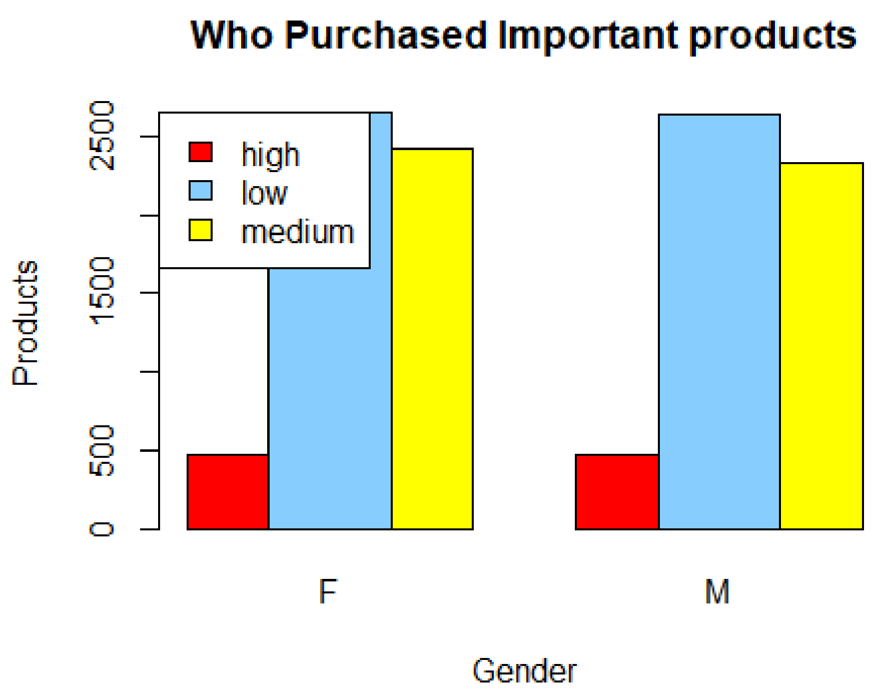


```
#display warehouse with different ratings
ware1<-table(data$Customer_rating,data$Warehouse_block)
```

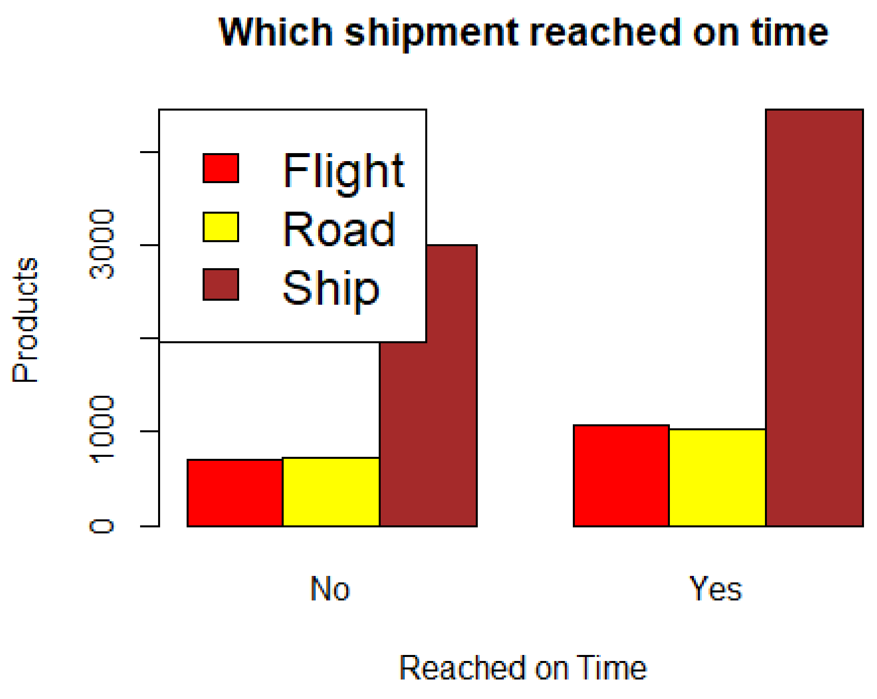
```
ratings <- c(1:5)
colors = c("green","orange","brown","yellow","red")
barplot(ware1, main = "Warehouse Rating",
      names.arg = ware$Var1, xlab ="Blocks", ylab = "Ratings", col = colors,
      legend.text = c(1,2,3,4,5), args.legend = list(cex=1,x = "topleft"))
```



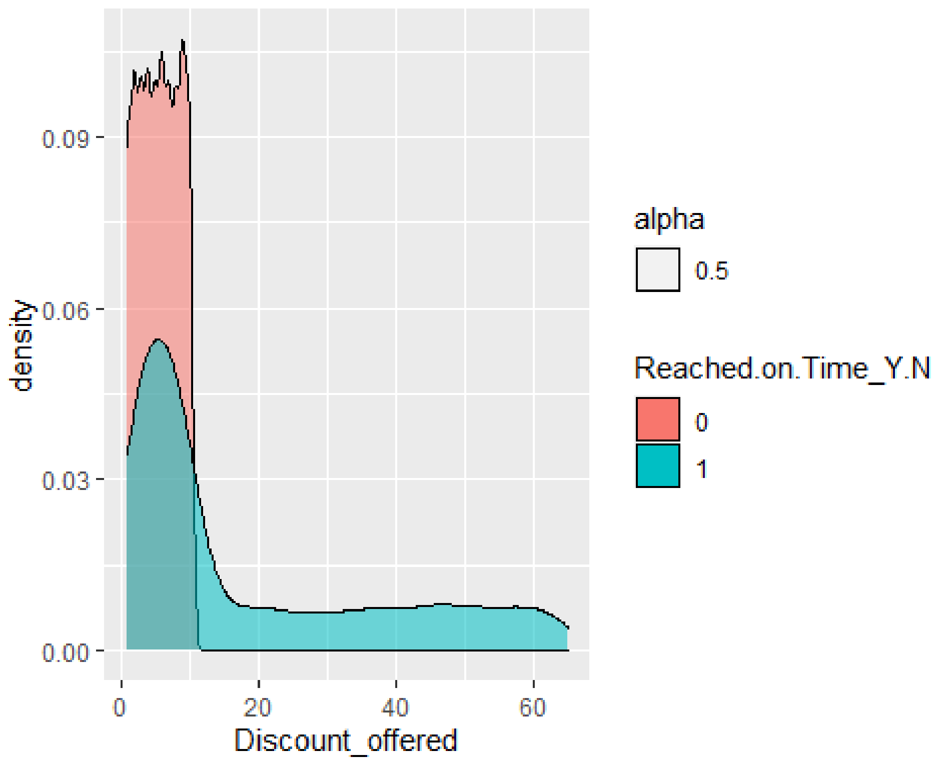
```
#display important product purchase
imp<-table(data$Product_importance,data$Gender)
barplot(imp,beside = TRUE,
      col = c("Red", "skyblue1","yellow"),
      main = "Who Purchased Important products",
      width=c(2,3), xlab = "Gender", ylab = "Products",
      legend.text = rownames(imp),
      args.legend = list(cex=1,x = "topleft"))
```



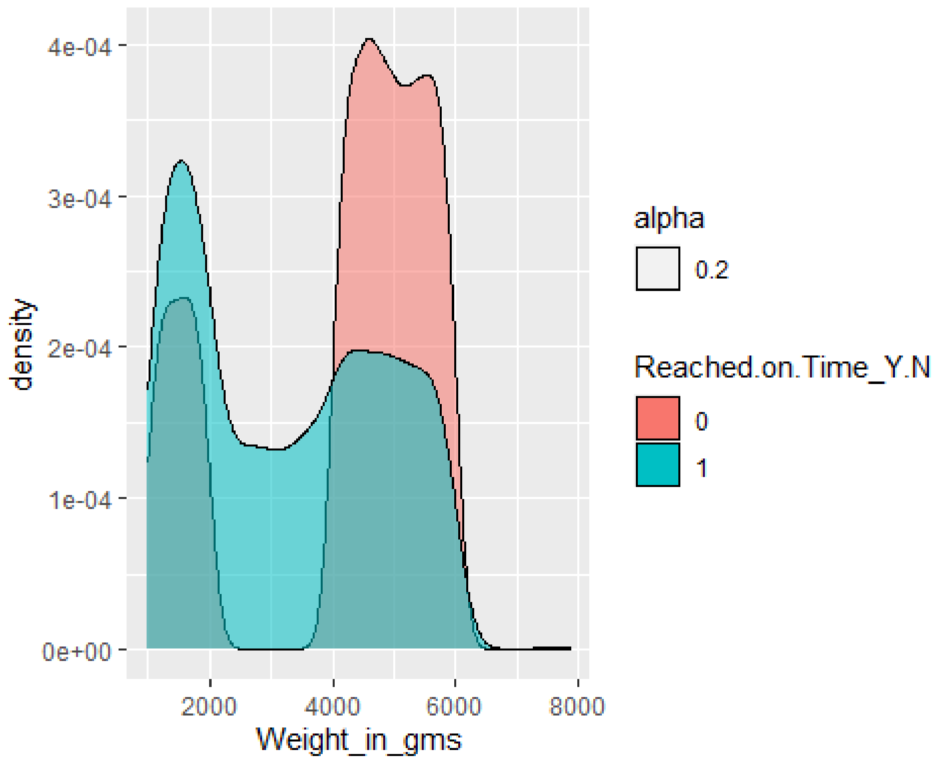
```
#display Which shipment reached on time
rtime<-data[,c(3,12)]
rtime<-table(rtime)
barplot(rtime,beside = TRUE,
    main = "Which shipment reached on time",
    col = c("red","yellow","brown"), xlab = "Reached on Time", ylab = "Products",
    legend.text = rownames(rtime), names.arg = c("No", "Yes"),
    args.legend = list(cex=1.5,x = "topleft"))
```



```
data1 <- data %>% mutate(Reached.on.Time_Y.N = as.factor(Reached.on.Time_Y.N))

# Display discount density graph
ggplot(data = data1) + geom_density(aes(fill = Reached.on.Time_Y.N, x = Discount_offered, y = ..density.., alpha=0.5))
```



```
# Display weight density graph
ggplot(data = data1) + geom_density(aes(fill = Reached.on.Time_Y.N, x = Weight_in_gms, y = ..density.., alpha=0.2))
```


density
4e-04
3e-04
2e-04
1e-04
0e+00
Weight_in_gms
alpha
Reached.on.Time_Y.N