Group Number 215: Black Friday Sales Prediction

First Name	Last Name	Email (hawk.iit.edu)	Student ID
Siddhi	Kulkarni	skulkarni13@hawk.iit.edu	A20430008
Pooja	Choudhari	cpooja@hawk.iit.edu	A20425832
Sagar	Ippili	sippili@hawk.iit.edu	A20417999

1. Import data set to R.

```
> data <- read.csv("BlackFriday.csv", sep=",", header=T)
> data[1:5,]
  User_ID Product_ID Gender  Age Occupation City_Category Stay_In_Current_City_Years
1 1000001  P00069042      F 0-17         10             A                          2
2 1000001  P00248942      F 0-17         10             A                          2
3 1000001  P00087842      F 0-17         10             A                          2
4 1000001  P00085442      F 0-17         10             A                          2
5 1000002  P00285442      M  55+         16             C                         4+
  Marital_Status Product_Category_1 Product_Category_2 Product_Category_3 Purchase
1              0                  3                 NA                 NA     8370
2              0                  1                  6                 14    15200
3              0                 12                 NA                 NA     1422
4              0                 12                 14                 NA     1057
5              0                  8                 NA                 NA     7969
> 
```

2. Replace the missing values in columns,
   a. **Product_Category_2:** Replacing the missing values with mean value of the column, i.e **9.84**.

```
> summary(data$Product_Category_2)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's 
   2.00    5.00    9.00    9.84   15.00   18.00  166986 
> data$Product_Category_2[is.na(data$Product_Category_2)] <- 9.84
> summary(data$Product_Category_2)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
  2.000   8.000   9.840   9.841  14.000  18.000 
```

   b. **Product_Category_3:** Replacing the missing values with mean value of the column, i.e **12.7**.

```
> summary(data$Product_Category_3)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's 
    3.0     9.0    14.0    12.7    16.0    18.0  373299 
> data$Product_Category_3[is.na(data$Product_Category_3)] <- 12.7
> summary(data$Product_Category_3)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
   3.00   12.70   12.70   12.69   12.70   18.00 
> 
```

3. Perform Z Test for

- Hypothesis 1:

```
> z.test(data$Product_Category_1,NULL,alternative = "less",mu = 5,sigma.x = sd(data$Product_Category_1),sigma.y = NULL,conf.level = 0.95)

	One-sample z-Test

data:  data$Product_Category_1
z = 57.774, p-value = 1
alternative hypothesis: true mean is less than 5
95 percent confidence interval:
       NA 5.303961
sample estimates:
mean of x
 5.295546
```

- Hypothesis 2:

```
> z.test(data$Product_Category_1,data$Product_Category_2,alternative="two.sided",
+ mu=0,sigma.x=sd(data$Product_Category_1),sigma.y=sd(data$Product_Category_2),conf.level=0.95)

	Two-sample z-Test

data:  data$Product_Category_1 and data$Product_Category_2
z = -590.05, p-value < 2.2e-16
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 -4.561032 -4.530831
sample estimates:
mean of x mean of y
 5.295546  9.841478
```

4. Ignoring the columns 'User_ID' and 'Product_ID, because they are unique to themselves and have no impact on our dependent variable **'Purchase'**, followed by creating a new data set for further building classification models.

```
> data<-subset(data,select = -User_ID)
> data<-subset(data,select = -Product_ID)
> data[1:3,]
  Gender  Age Occupation City_Category Stay_In_Current_City_Years Marital_Status
1      F 0-17         10             A                          2              0
2      F 0-17         10             A                          2              0
3      F 0-17         10             A                          2              0
  Product_Category_1 Product_Category_2 Product_Category_3 Purchase
1                  3               9.84               12.7     8370
2                  1               6.00               14.0    15200
3                 12               9.84               12.7     1422
> 
```

5. Building Naïve-Bayes Classification Model:
   a. Clone the data set to a new variable, 'data_nb' to use for building Naïve-Bayes classification model.

```
> data_nb<-data
> data_nb[1:5,]
  Gender   Age Occupation City_Category Stay_In_Current_City_Years Marital_Status
1      F  0-17         10             A                          2              0
2      F  0-17         10             A                          2              0
3      F  0-17         10             A                          2              0
4      F  0-17         10             A                          2              0
5      M   55+         16             C                         4+              0
  Product_Category_1 Product_Category_2 Product_Category_3 Purchase
1                  3               9.84               12.7     8370
2                  1               6.00               14.0    15200
3                 12               9.84               12.7     1422
4                 12              14.00               12.7     1057
5                  8               9.84               12.7     7969
>
```

b. Transform the '**Purchase**' variable into a dummy variable which has two values, i.e **1** and **0**. Here, **1** means higher purchase [i.e >Mean Value] amount and **0** means lower purchase [i.e <Mean Value] amount made by a customer.

```
> summary(data_nb$Purchase)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
    185    5866    8062    9334   12073   23961
> data_nb$Purchase<-ifelse(data_nb$Purchase>9334,1,0)
> data_nb$Purchase<-as.factor(data_nb$Purchase)
> head(data_nb$Purchase)
[1] 0 1 0 0 0 1
Levels: 0 1
>
```

c. Group the numerical variables using the **cut()**, i.e

I. 'Product_Category_1':

```
> summary(data_nb$Product_Category_1)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  1.000   1.000   5.000   5.296   8.000  18.000
> data_nb$Product_Category_1<-cut(data_nb$Product_Category_1,3)
> head(data_nb$Product_Category_1)
[1] (0.983,6.67] (0.983,6.67] (6.67,12.3]  (6.67,12.3]  (6.67,12.3]  (0.983,6.67]
Levels: (0.983,6.67] (6.67,12.3] (12.3,18]
```

II. 'Product_Category_2':

```
> summary(data_nb$Product_Category_2)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  2.000   8.000   9.840   9.841  14.000  18.000
> data_nb$Product_Category_2<-cut(data_nb$Product_Category_2,4)
> head(data_nb$Product_Category_2)
[1] (6,10]    (1.98,6] (6,10]    (10,14]   (6,10]    (1.98,6]
Levels: (1.98,6] (6,10] (10,14] (14,18]
>
```

III. 'Product_Category_3':

```
> summary(data_nb$Product_Category_3)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   3.00   12.70   12.70   12.69   12.70   18.00
> data_nb$Product_Category_3<-cut(data_nb$Product_Category_3,4)
> head(data_nb$Product_Category_3)
[1] (10.5,14.2] (10.5,14.2] (10.5,14.2] (10.5,14.2] (10.5,14.2] (10.5,14.2]
Levels: (2.98,6.75] (6.75,10.5] (10.5,14.2] (14.2,18]
> 
```

d. Check the entire data set after grouping

```
> head(data_nb)
  Gender   Age Occupation City_Category Stay_In_Current_City_Years Marital_Status
1      F  0-17         10             A                          2              0
2      F  0-17         10             A                          2              0
3      F  0-17         10             A                          2              0
4      F  0-17         10             A                          2              0
5      M   55+         16             C                         4+              0
6      M 26-35         15             A                          3              0
  Product_Category_1 Product_Category_2 Product_Category_3 Purchase
1       (0.983,6.67]             (6,10]        (10.5,14.2]        0
2       (0.983,6.67]           (1.98,6]        (10.5,14.2]        1
3        (6.67,12.3]             (6,10]        (10.5,14.2]        0
4        (6.67,12.3]            (10,14]        (10.5,14.2]        0
5        (6.67,12.3]             (6,10]        (10.5,14.2]        0
6       (0.983,6.67]           (1.98,6]        (10.5,14.2]        1
> 
```

e. Dataset being very large (approx. 500 thousand records), we will split data into train data and test data for hold-out evaluation later.

```
> index.data_nb<-sample(1:nrow(data_nb),size=round(0.8*nrow(data_nb)))
> train.data_nb<-data_nb[index.data_nb,]
> test.data_nb<-data_nb[-index.data_nb,]
> head(train.data_nb)
       Gender   Age Occupation City_Category Stay_In_Current_City_Years Marital_Status
170762      M 36-45          2             A                          2              0
11935       M 26-35          0             C                          1              0
305395      M 46-50         12             C                          1              1
487998      M 36-45         17             C                          3              0
159734      M 18-25          0             B                          1              0
390377      F 46-50         16             A                          1              1
       Product_Category_1 Product_Category_2 Product_Category_3 Purchase
170762         (6.67,12.3]            (10,14]        (10.5,14.2]        0
11935         (0.983,6.67]            (14,18]        (10.5,14.2]        1
305395         (6.67,12.3]             (6,10]        (10.5,14.2]        0
487998         (6.67,12.3]             (6,10]        (10.5,14.2]        0
159734        (0.983,6.67]             (6,10]        (10.5,14.2]        1
390377        (0.983,6.67]            (10,14]        (10.5,14.2]        0
> head(tes.data_nb)
Error in head(tes.data_nb) : object 'tes.data_nb' not found
> head(test.data_nb)
   Gender   Age Occupation City_Category Stay_In_Current_City_Years Marital_Status
2       F  0-17         10             A                          2              0
4       F  0-17         10             A                          2              0
19      M 36-45          1             B                          1              1
21      M 26-35         12             C                         4+              1
28      M 26-35         17             C                          0              0
30      F 36-45          1             B                         4+              1
   Product_Category_1 Product_Category_2 Product_Category_3 Purchase
2        (0.983,6.67]           (1.98,6]        (10.5,14.2]        1
4         (6.67,12.3]            (10,14]        (10.5,14.2]        0
19       (0.983,6.67]            (10,14]          (14.2,18]        1
21       (0.983,6.67]            (10,14]        (10.5,14.2]        0
28       (0.983,6.67]            (10,14]        (10.5,14.2]        0
30       (0.983,6.67]           (1.98,6]        (6.75,10.5]        1
> 
```

f. Build the Naïve-Bayes classification model.

```
> model_nb<-naive_bayes(Purchase~.,train.data_nb)
> pred_nb<-predict(model_nb,test.data_nb)
> head(predict(model_nb,test.data_nb,type="prob"))
             0          1
[1,] 0.4773295 0.52267053
[2,] 0.9418721 0.05812791
[3,] 0.5076089 0.49239106
[4,] 0.7416757 0.25832425
[5,] 0.7383431 0.26165689
[6,] 0.1365317 0.86346833
> library(Metrics)
> accuracy(test.data_nb$Purchase,pred_nb)
[1] 0.7089057
> 
```

From the Naïve-Bayes model, we can see that the accuracy is **0.7089057**.

g. Try building different Naïve-Bayes classification models by categorizing the numerical variables in to different number of groups using the 'cut()'.

I. Product_Category_1 with 3, Product_Category_2 with 2 and Product_Category_3 with 4 groups each.

```
> data_nb<-data
> data_nb<-subset(data,select = -User_ID)
> data_nb<-subset(data_nb,select = -Product_ID)
> data_nb$Purchase<-ifelse(data_nb$Purchase>9334,1,0)
> data_nb$Purchase<-as.factor(data_nb$Purchase)
> head(data_nb)
  Gender   Age Occupation City_Category Stay_In_Current_City_Years Marital_Status
1      F  0-17         10             A                          2              0
2      F  0-17         10             A                          2              0
3      F  0-17         10             A                          2              0
4      F  0-17         10             A                          2              0
5      M   55+         16             C                         4+              0
6      M 26-35         15             A                          3              0
> data_nb$Occupation<-as.factor(data$Occupation)
> data_nb$Marital_Status<-as.factor(data_nb$Marital_Status)
> data_nb$Product_Category_1<-cut(data_nb$Product_Category_1,3)
> data_nb$Product_Category_2<-cut(data_nb$Product_Category_2,2)
> data_nb$Product_Category_3<-cut(data_nb$Product_Category_3,4)
> index.data_nb<-sample(1:nrow(data_nb),size=round(0.8*nrow(data_nb)))
> train.data_nb<-data_nb[index.data_nb,]
> test.data_nb<-data_nb[-index.data_nb,]
> model_nb<-naive_bayes(Purchase~.,train.data_nb)
> pred<-predict(model_nb,test.data_nb)
> head(predict(model_nb,test.data_nb,type="prob"))
             0         1
[1,] 0.2617788 0.7382212
[2,] 0.6622482 0.3377518
[3,] 0.8389146 0.1610854
[4,] 0.7349300 0.2650700
[5,] 0.3308822 0.6691178
[6,] 0.7090515 0.2909485
> accuracy(test.data_nb$Purchase,pred)
[1] 0.6986467
```

From the above Naïve-Bayes model, we can see that the accuracy is 0.6986467.

II. Product_Category_1 with 6, Product_Category_2 with 6 and Product_Category_3 with 2 groups each.

```
> data_nb<-data_nb_new
> data_nb$Product_Category_2<-cut(data_nb$Product_Category_2,6)
> data_nb$Product_Category_3<-cut(data_nb$Product_Category_3,6)
> data_nb$Product_Category_1<-cut(data_nb$Product_Category_1,2)
> index.data_nb<-sample(1:nrow(data_nb),size=round(0.8*nrow(data_nb)))
> train.data_nb<-data_nb[index.data_nb,]
> test.data_nb<-data_nb[-index.data_nb,]
> model_nb<-naive_bayes(Purchase~.,train.data_nb)
> pred<-predict(model_nb,test.data_nb)
> accuracy(test.data_nb$Purchase,pred)
[1] 0.7030089
```

From the above Naïve-Bayes model, we can see that the accuracy is 0.7030089.

III. **Product_Category_1 with 8, Product_Category_2 with 8 and Product_Category_3 with 6 groups each.**

```
> data_nb<-data_nb_new
> data_nb$Product_Category_3<-cut(data_nb$Product_Category_3,8)
> data_nb$Product_Category_2<-cut(data_nb$Product_Category_2,8)
> data_nb$Product_Category_1<-cut(data_nb$Product_Category_1,6)
> index.data_nb<-sample(1:nrow(data_nb),size=round(0.8*nrow(data_nb)))
> train.data_nb<-data_nb[index.data_nb,]
> test.data_nb<-data_nb[-index.data_nb,]
> model_nb<-naive_bayes(Purchase~.,train.data_nb)
> pred<-predict(model_nb,test.data_nb)
> accuracy(test.data_nb$Purchase,pred)
[1] 0.7899735
```

From the above Naïve-Bayes model, we can see that the accuracy is 0.7899735.

IV. **Product_Category_1 with 8, Product_Category_2 with 8 and Product_Category_3 with 8 groups each.**

```
> data_nb<-data_nb_new
> data_nb$Product_Category_3<-cut(data_nb$Product_Category_3,8)
> data_nb$Product_Category_2<-cut(data_nb$Product_Category_2,8)
> data_nb$Product_Category_1<-cut(data_nb$Product_Category_1,8)
> index.data_nb<-sample(1:nrow(data_nb),size=round(0.8*nrow(data_nb)))
> train.data_nb<-data_nb[index.data_nb,]
> test.data_nb<-data_nb[-index.data_nb,]
> model_nb<-naive_bayes(Purchase~.,train.data_nb)
> pred<-predict(model_nb,test.data_nb)
> accuracy(test.data_nb$Purchase,pred)
[1] 0.8362275
```

From the above Naïve-Bayes model, we can see that the accuracy is 0.8362275.

From the above built different Naïve-Bayes models, we can achieve a highest accuracy of **0.8362275**.

Hence, we can consider this model as the best among the Naïve-Bayes Classification models built above.

6. Building KNN Classification Model:
   a. Clone the data set to a new variable, 'data_knn' to use for building KNN classification model.

```
> data_knn<-data
> data_knn[1:5,]
  Gender  Age Occupation City_Category Stay_In_Current_City_Years Marital_Status
1      F 0-17         10             A                          2              0
2      F 0-17         10             A                          2              0
3      F 0-17         10             A                          2              0
4      F 0-17         10             A                          2              0
5      M  55+         16             C                         4+              0
  Product_Category_1 Product_Category_2 Product_Category_3 Purchase
1                  3               9.84               12.7     8370
2                  1               6.00               14.0    15200
3                 12               9.84               12.7     1422
4                 12              14.00               12.7     1057
5                  8               9.84               12.7     7969
> 
```

b. Transform the '**Purchase**' variable into a dummy variable which has two values, i.e **1** and **0**. Here, **1** means higher purchase [i.e >Mean Value] amount and **0** means lower purchase [i.e <Mean Value] amount made by a customer.

```
> summary(data_knn$Purchase)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
    185    5866    8062    9334   12073   23961 
> data_knn$Purchase<-ifelse(data_knn$Purchase>9334,1,0)
> data_knn$Purchase<-as.factor(data_knn$Purchase)
> head(data_knn$Purchase)
[1] 0 1 0 0 0 1
Levels: 0 1
> 
```

c. Create the dummy variables to the categorical variables 'Gender', 'Age', 'Occupation', 'City_Category', 'Stay_In_Current_City_Years', 'Marital_Status'.

```
> library(dummies)
dummies-1.5.6 provided by Decision Patterns

> data_knn<-dummy.data.frame(data_knn,names=c("Gender","Age","Occupation","City_Category","Stay_In_Current_City_Years","Marital_Status"))
> head(data_knn)
  GenderF GenderM Age0-17 Age18-25 Age26-35 Age36-45 Age46-50 Age51-55 Age55+ Occupation0 Occupation1 Occupation2 Occupation3 Occupation4
1       1       0       1        0        0        0        0        0      0           0           0           0           0           0
2       1       0       1        0        0        0        0        0      0           0           0           0           0           0
3       1       0       1        0        0        0        0        0      0           0           0           0           0           0
4       1       0       1        0        0        0        0        0      0           0           0           0           0           0
5       0       1       0        0        0        0        0        0      1           0           0           0           0           0
6       0       1       0        0        1        0        0        0      0           0           0           0           0           0
  Occupation7 Occupation8 Occupation9 Occupation10 Occupation11 Occupation12 Occupation13 Occupation14 Occupation15 Occupation16 Occupati
1           0           0           0            1            0            0            0            0            0            0
2           0           0           0            1            0            0            0            0            0            0
3           0           0           0            1            0            0            0            0            0            0
4           0           0           0            1            0            0            0            0            0            0
5           0           0           0            0            0            0            0            0            0            1
6           0           0           0            0            0            0            0            0            1            0
  Occupation19 Occupation20 City_CategoryA City_CategoryB City_CategoryC Stay_In_Current_City_Years0 Stay_In_Current_City_Years1 Stay_In_(
1            0            0              1              0              0                           0                           0
2            0            0              1              0              0                           0                           0
3            0            0              1              0              0                           0                           0
4            0            0              1              0              0                           0                           0
5            0            0              0              0              1                           0                           0
```

d. Extract numerical variables and normalize the selected data [scaling].

```
> numeric.vars.knn = sapply(data_knn,is.numeric)
> data_knn[numeric.vars.knn] <- lapply(data_knn[numeric.vars.knn],scale)
> head(data_knn)
     GenderF    GenderM    Age0-17    Age18-25   Age26-35   Age36-45   Age46-50
1  1.7511363 -1.7511363  5.9625827 -0.4710879 -0.8154179 -0.4999519 -0.3005111
2  1.7511363 -1.7511363  5.9625827 -0.4710879 -0.8154179 -0.4999519 -0.3005111
3  1.7511363 -1.7511363  5.9625827 -0.4710879 -0.8154179 -0.4999519 -0.3005111
4  1.7511363 -1.7511363  5.9625827 -0.4710879 -0.8154179 -0.4999519 -0.3005111
5 -0.5710567  0.5710567 -0.1677122 -0.4710879 -0.8154179 -0.4999519 -0.3005111
6 -0.5710567  0.5710567 -0.1677122 -0.4710879  1.2263628 -0.4999519 -0.3005111
  Occupation5 Occupation6 Occupation7 Occupation8 Occupation9 Occupation10 Occu
1   -0.151006  -0.1956641  -0.3471115 -0.05331976  -0.1076025    6.4488003   -(
2   -0.151006  -0.1956641  -0.3471115 -0.05331976  -0.1076025    6.4488003   -(
3   -0.151006  -0.1956641  -0.3471115 -0.05331976  -0.1076025    6.4488003   -(
4   -0.151006  -0.1956641  -0.3471115 -0.05331976  -0.1076025    6.4488003   -(
5   -0.151006  -0.1956641  -0.3471115 -0.05331976  -0.1076025   -0.1550673   -(
6   -0.151006  -0.1956641  -0.3471115 -0.05331976  -0.1076025   -0.1550673   -(
  Occupation18 Occupation19 Occupation20 City_CategoryA City_CategoryB City_Cat
1   -0.1108463   -0.1256246   -0.2553648      1.6482419     -0.8532733     -0.
2   -0.1108463   -0.1256246   -0.2553648      1.6482419     -0.8532733     -0.
3   -0.1108463   -0.1256246   -0.2553648      1.6482419     -0.8532733     -0.
4   -0.1108463   -0.1256246   -0.2553648      1.6482419     -0.8532733     -0.
5   -0.1108463   -0.1256246   -0.2553648     -0.6067059     -0.8532733      1.
6   -0.1108463   -0.1256246   -0.2553648      1.6482419     -0.8532733     -0.
  Stay_In_Current_City_Years2 Stay_In_Current_City_Years3 Stay_In_Current_City_
1                   2.0988099                  -0.4582973                    -(
2                   2.0988099                  -0.4582973                    -(
3                   2.0988099                  -0.4582973                    -(
4                   2.0988099                  -0.4582973                    -(
5                  -0.4764596                  -0.4582973                     :
6                  -0.4764596                   2.1819857                    -(
  Product_Category_3 Purchase
1        0.004042377        0
2        0.574222766        1
3        0.004042377        0
```

e. Dataset being very large (approx. 500 thousand records), we will split data into train data and test data for hold-out evaluation later.

```
> set.seed(123)
> test.index<-1:107515
> train.data_knn<-data_knn[-test.index,]
> test.data_knn<-data_knn[test.index,]
> train.Purchase<-data_knn$Purchase[-test.index]
> test.Purchase<-data_knn$Purchase[test.index]
> head(train.data_knn)
          GenderF   GenderM    Age0-17   Age18-25   Age26-35
107516 -0.5710567 0.5710567 -0.1677122 -0.4710879 -0.8154179
107517 -0.5710567 0.5710567 -0.1677122 -0.4710879 -0.8154179
107518 -0.5710567 0.5710567 -0.1677122 -0.4710879  1.2263628
107519 -0.5710567 0.5710567 -0.1677122 -0.4710879  1.2263628
107520 -0.5710567 0.5710567 -0.1677122 -0.4710879  1.2263628
107521 -0.5710567 0.5710567 -0.1677122 -0.4710879  1.2263628
       Occupation5 Occupation6 Occupation7 Occupation8 Occupa
107516   -0.151006  -0.1956641   2.8809129 -0.05331976  -0.10
107517   -0.151006  -0.1956641   2.8809129 -0.05331976  -0.10
107518   -0.151006  -0.1956641  -0.3471115 -0.05331976  -0.10
107519   -0.151006  -0.1956641  -0.3471115 -0.05331976  -0.10
107520   -0.151006  -0.1956641  -0.3471115 -0.05331976  -0.10
107521   -0.151006  -0.1956641  -0.3471115 -0.05331976  -0.10
       Occupation17 Occupation18 Occupation19 Occupation20 Ci
107516   -0.2800306   -0.1108463   -0.1256246   -0.2553648
107517   -0.2800306   -0.1108463   -0.1256246   -0.2553648
107518   -0.2800306   -0.1108463   -0.1256246   -0.2553648
107519   -0.2800306   -0.1108463   -0.1256246   -0.2553648
107520   -0.2800306   -0.1108463   -0.1256246   -0.2553648
> head(test.data_knn)
     GenderF    GenderM    Age0-17   Age18-25   Age26-35    Age
1  1.7511363 -1.7511363  5.9625827 -0.4710879 -0.8154179 -0.49
2  1.7511363 -1.7511363  5.9625827 -0.4710879 -0.8154179 -0.49
3  1.7511363 -1.7511363  5.9625827 -0.4710879 -0.8154179 -0.49
4  1.7511363 -1.7511363  5.9625827 -0.4710879 -0.8154179 -0.49
5 -0.5710567  0.5710567 -0.1677122 -0.4710879 -0.8154179 -0.49
6 -0.5710567  0.5710567 -0.1677122 -0.4710879  1.2263628 -0.49
  Occupation5 Occupation6 Occupation7 Occupation8 Occupation9
1   -0.151006  -0.1956641  -0.3471115 -0.05331976  -0.1076025
2   -0.151006  -0.1956641  -0.3471115 -0.05331976  -0.1076025
3   -0.151006  -0.1956641  -0.3471115 -0.05331976  -0.1076025
4   -0.151006  -0.1956641  -0.3471115 -0.05331976  -0.1076025
5   -0.151006  -0.1956641  -0.3471115 -0.05331976  -0.1076025
6   -0.151006  -0.1956641  -0.3471115 -0.05331976  -0.1076025
  Occupation18 Occupation19 Occupation20 City_CategoryA City_C
1   -0.1108463   -0.1256246   -0.2553648      1.6482419     -0
2   -0.1108463   -0.1256246   -0.2553648      1.6482419     -0
3   -0.1108463   -0.1256246   -0.2553648      1.6482419     -0
4   -0.1108463   -0.1256246   -0.2553648      1.6482419     -0
5   -0.1108463   -0.1256246   -0.2553648     -0.6067059     -0
6   -0.1108463   -0.1256246   -0.2553648      1.6482419     -0
  Stay_In_Current_City_Years2 Stay_In_Current_City_Years3 Stay
1                   2.0988099                  -0.4582973
2                   2.0988099                  -0.4582973
3                   2.0988099                  -0.4582973
4                   2.0988099                  -0.4582973
```

f. Calculate the Accuracy of the KNN Classification models and find the best model.

```
> library(Metrics)
> accuracy(test.Purchase,knn.1)
[1] 0.9862159
> knn.5<-knn(train.data_knn,test.data_knn,train.Purchase,k=5)
> accuracy(test.Purchase,knn.5)
[1] 0.9556248
> knn.15<-knn(train.data_knn,test.data_knn,train.Purchase,k=15)
> accuracy(test.Purchase,knn.15)
[1] 0.910366
```

The largest accuracy of the KNN model is **0.9862** at **K=1**.

7. Building Logistic Regression Model
   a. Clone the data set to a new variable, 'data_lg' to use for building Logistic Regression model

```
> data_lg<-data
> data_lg[1:5,]
  Gender  Age Occupation City_Category Stay_In_Current_City_Years Marital_Status
1      F 0-17         10             A                          2              0
2      F 0-17         10             A                          2              0
3      F 0-17         10             A                          2              0
4      F 0-17         10             A                          2              0
5      M  55+         16             C                         4+              0
  Product_Category_1 Product_Category_2 Product_Category_3 Purchase
1                  3               9.84               12.7     8370
2                  1               6.00               14.0    15200
3                 12               9.84               12.7     1422
4                 12              14.00               12.7     1057
5                  8               9.84               12.7     7969
> |
```

   b. Transform the '**Purchase**' variable into a dummy variable which has two values, i.e **1** and **0**. Here, **1** means higher purchase [i.e >Mean Value] amount and **0** means lower purchase [i.e <Mean Value] amount made by a customer.

```
> summary(data_lg$Purchase)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
    185    5866    8062    9334   12073   23961
> data_lg$Purchase<-ifelse(data_lg$Purchase>9334,1,0)
> data_lg$Purchase<-as.factor(data_lg$Purchase)
> head(data_lg$Purchase)
[1] 0 1 0 0 0 1
Levels: 0 1
> |
```

   c. Split data into train data and test data.

```
> train.index<-createDataPartition(data_lg$Purchase,p=0.8,list=FALSE)
> train.data_lg<-data_lg[train.index,]
> test.data_lg<-data_lg[-train.index,]
> head(train.data_lg)
  Gender   Age Occupation City_Category Stay_In_Current_City_Years Marital_Status
1      F  0-17         10             A                          2              0
3      F  0-17         10             A                          2              0
4      F  0-17         10             A                          2              0
6      M 26-35         15             A                          3              0
7      M 46-50          7             B                          2              1
8      M 46-50          7             B                          2              1
  Product_Category_1 Product_Category_2 Product_Category_3 Purchase
1                  3               9.84               12.7        0
3                 12               9.84               12.7        0
4                 12              14.00               12.7        0
6                  1               2.00               12.7        1
7                  1               8.00               17.0        1
8                  1              15.00               12.7        1
> head(test.data_lg)
   Gender   Age Occupation City_Category Stay_In_Current_City_Years Marital_Status
2       F  0-17         10             A                          2              0
5       M   55+         16             C                         4+              0
14      M 26-35         20             A                          1              1
22      M 26-35         12             C                         4+              1
26      M 26-35         17             C                          0              0
30      F 36-45          1             B                         4+              1
   Product_Category_1 Product_Category_2 Product_Category_3 Purchase
2                   1               6.00               14.0        1
5                   8               9.84               12.7        0
14                  1               2.00                5.0        1
22                  8               9.84               12.7        1
26                  6               8.00               12.7        1
30                  2               4.00                8.0        1
> 
```

d. Build full logistic model using ‘glm()’.

```
> model_lg_full<-glm(as.factor(Purchase)~.,data=train.data_lg,family=binomial())
> summary(model_lg_full)

Call:
glm(formula = as.factor(Purchase) ~ ., family = binomial(), data = train.data_lg)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.6622  -0.9797  -0.7115   1.0496   2.4619

Coefficients:
                               Estimate Std. Error  z value Pr(>|z|)
(Intercept)                   0.3255572  0.0291102   11.184  < 2e-16 ***
GenderM                       0.2224393  0.0078314   28.404  < 2e-16 ***
Age18-25                      0.1153657  0.0215789    5.346 8.98e-08 ***
Age26-35                      0.1945145  0.0209692    9.276  < 2e-16 ***
Age36-45                      0.2572174  0.0215513   11.935  < 2e-16 ***
Age46-50                      0.2499432  0.0236833   10.554  < 2e-16 ***
Age51-55                      0.3814411  0.0241499   15.795  < 2e-16 ***
Age55+                        0.3456951  0.0264662   13.062  < 2e-16 ***
Occupation                    0.0023126  0.0005136    4.502 6.72e-06 ***
City_CategoryB                0.0651329  0.0082552    7.890 3.02e-15 ***
City_CategoryC                0.2675380  0.0088579   30.203  < 2e-16 ***
Stay_In_Current_City_Years1   0.0159197  0.0106084    1.501 0.133440
Stay_In_Current_City_Years2   0.0184713  0.0118245    1.562 0.118262
Stay_In_Current_City_Years3  -0.0022555  0.0120262   -0.188 0.851233
Stay_In_Current_City_Years4+  0.0322954  0.0123173    2.622 0.008743 **
Marital_Status               -0.0245683  0.0071527   -3.435 0.000593 ***
Product_Category_1           -0.1728760  0.0011102 -155.722  < 2e-16 ***
Product_Category_2           -0.0486791  0.0008902  -54.686  < 2e-16 ***
Product_Category_3            0.0073419  0.0014769    4.971 6.65e-07 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 579321  on 430061  degrees of freedom
Residual deviance: 529299  on 430043  degrees of freedom
AIC: 529337

Number of Fisher Scoring iterations: 4

>
```

e. Build a base model with one x-variable.

```
> model_lg_base<-glm(as.factor(Purchase)~Gender,data=train.data_lg,family=binomial())
> summary(model_lg_base)

Call:
glm(formula = as.factor(Purchase) ~ Gender, family = binomial(),
    data = train.data_lg)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.0405  -1.0405  -0.9281   1.3208   1.4491

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) -0.619282   0.006449  -96.03   <2e-16 ***
GenderM      0.288398   0.007366   39.15   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 579321  on 430061  degrees of freedom
Residual deviance: 577765  on 430060  degrees of freedom
AIC: 577769

Number of Fisher Scoring iterations: 4
```

f. Build a stepwise forward model.

```
> model_lg_fwd<-step(model_lg_base,scope=list(upper=model_lg_full,lower=~1),direction="forward",trace=T)
Start:  AIC=577769.4
as.factor(Purchase) ~ Gender

                           Df Deviance    AIC
+ Product_Category_1        1   534227 534233
+ Product_Category_2        1   559375 559381
+ City_Category             2   576216 576224
+ Product_Category_3        1   576723 576729
+ Age                       6   577612 577628
+ Occupation                1   577699 577705
+ Stay_In_Current_City_Years  4   577739 577751
<none>                          577765 577769
+ Marital_Status            1   577765 577771

Step:  AIC=534232.8
as.factor(Purchase) ~ Gender + Product_Category_1

                           Df Deviance    AIC
+ Product_Category_2        1   531142 531150
+ City_Category             2   533015 533025
+ Age                       6   533594 533612
+ Product_Category_3        1   534066 534074
+ Occupation                1   534169 534177
+ Marital_Status            1   534205 534213
+ Stay_In_Current_City_Years  4   534203 534217
<none>                          534227 534233

Step:  AIC=531149.6
as.factor(Purchase) ~ Gender + Product_Category_1 + Product_Category_2

                           Df Deviance    AIC
+ City_Category             2   529914 529926
+ Age                       6   530434 530454
+ Occupation                1   531080 531090
+ Product_Category_3        1   531112 531122
+ Marital_Status            1   531117 531127
+ Stay_In_Current_City_Years  4   531116 531132
```

```
<none>                          531142 531150

Step:  AIC=529926.4
as.factor(Purchase) ~ Gender + Product_Category_1 + Product_Category_2 +
    City_Category

                             Df Deviance    AIC
+ Age                         6   529367 529391
+ Occupation                  1   529871 529885
+ Product_Category_3          1   529885 529899
+ Marital_Status              1   529901 529915
+ Stay_In_Current_City_Years  4   529899 529919
<none>                            529914 529926

Step:  AIC=529391.4
as.factor(Purchase) ~ Gender + Product_Category_1 + Product_Category_2 +
    City_Category + Age

                             Df Deviance    AIC
+ Product_Category_3          1   529343 529369
+ Occupation                  1   529347 529373
+ Marital_Status              1   529356 529382
+ Stay_In_Current_City_Years  4   529355 529387
<none>                            529367 529391

Step:  AIC=529368.6
as.factor(Purchase) ~ Gender + Product_Category_1 + Product_Category_2 +
    City_Category + Age + Product_Category_3

                             Df Deviance    AIC
+ Occupation                  1   529322 529350
+ Marital_Status              1   529331 529359
+ Stay_In_Current_City_Years  4   529331 529365
<none>                            529343 529369

Step:  AIC=529350.1
as.factor(Purchase) ~ Gender + Product_Category_1 + Product_Category_2 +
    City_Category + Age + Product_Category_3 + Occupation


                             Df Deviance    AIC
+ Marital_Status              1   529310 529340
+ Stay_In_Current_City_Years  4   529310 529346
<none>                            529322 529350

Step:  AIC=529340.3
as.factor(Purchase) ~ Gender + Product_Category_1 + Product_Category_2 +
    City_Category + Age + Product_Category_3 + Occupation + Marital_Status

                             Df Deviance    AIC
+ Stay_In_Current_City_Years  4   529299 529337
<none>                            529310 529340

Step:  AIC=529336.5
as.factor(Purchase) ~ Gender + Product_Category_1 + Product_Category_2 +
    City_Category + Age + Product_Category_3 + Occupation + Marital_Status +
    Stay_In_Current_City_Years
```

```
> summary(model_lg_fwd)

Call:
glm(formula = as.factor(Purchase) ~ Gender + Product_Category_1 +
    Product_Category_2 + City_Category + Age + Product_Category_3 +
    Occupation + Marital_Status + Stay_In_Current_City_Years,
    family = binomial(), data = train.data_lg)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.6622  -0.9797  -0.7115   1.0496   2.4619

Coefficients:
                               Estimate Std. Error  z value Pr(>|z|)
(Intercept)                   0.3255572  0.0291102   11.184  < 2e-16 ***
GenderM                       0.2224393  0.0078314   28.404  < 2e-16 ***
Product_Category_1           -0.1728760  0.0011102 -155.722  < 2e-16 ***
Product_Category_2           -0.0486791  0.0008902  -54.686  < 2e-16 ***
City_CategoryB                0.0651329  0.0082552    7.890 3.02e-15 ***
City_CategoryC                0.2675380  0.0088579   30.203  < 2e-16 ***
Age18-25                      0.1153657  0.0215789    5.346 8.98e-08 ***
Age26-35                      0.1945145  0.0209692    9.276  < 2e-16 ***
Age36-45                      0.2572174  0.0215513   11.935  < 2e-16 ***
Age46-50                      0.2499432  0.0236833   10.554  < 2e-16 ***
Age51-55                      0.3814411  0.0241499   15.795  < 2e-16 ***
Age55+                        0.3456951  0.0264662   13.062  < 2e-16 ***
Product_Category_3            0.0073419  0.0014769    4.971 6.65e-07 ***
Occupation                    0.0023126  0.0005136    4.502 6.72e-06 ***
Marital_Status               -0.0245683  0.0071527   -3.435 0.000593 ***
Stay_In_Current_City_Years1   0.0159197  0.0106084    1.501 0.133440
Stay_In_Current_City_Years2   0.0184713  0.0118245    1.562 0.118262
Stay_In_Current_City_Years3  -0.0022555  0.0120262   -0.188 0.851233
Stay_In_Current_City_Years4+  0.0322954  0.0123173    2.622 0.008743 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 579321  on 430061  degrees of freedom
Residual deviance: 529299  on 430043  degrees of freedom
AIC: 529337

Number of Fisher Scoring iterations: 4

>
```

g. Build a stepwise backward model using the full model built previously.

```
> model_lg_bwd<-step(model_lg_full,direction="backward",trace=T)
Start:  AIC=529336.5
as.factor(Purchase) ~ Gender + Age + Occupation + City_Category +
    Stay_In_Current_City_Years + Marital_Status + Product_Category_1 +
    Product_Category_2 + Product_Category_3

                             Df Deviance    AIC
<none>                            529299 529337
- Stay_In_Current_City_Years  4   529310 529340
- Marital_Status              1   529310 529346
- Occupation                  1   529319 529355
- Product_Category_3          1   529323 529359
- Age                         6   529816 529842
- Gender                      1   530112 530148
- City_Category               2   530352 530386
- Product_Category_2          1   532321 532357
- Product_Category_1          1   557414 557450
> summary(model_lg_bwd)

Call:
glm(formula = as.factor(Purchase) ~ Gender + Age + Occupation +
    City_Category + Stay_In_Current_City_Years + Marital_Status +
    Product_Category_1 + Product_Category_2 + Product_Category_3,
    family = binomial(), data = train.data_lg)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.6622  -0.9797  -0.7115   1.0496   2.4619

Coefficients:
                              Estimate Std. Error  z value Pr(>|z|)
(Intercept)                  0.3255572  0.0291102   11.184  < 2e-16 ***
GenderM                      0.2224393  0.0078314   28.404  < 2e-16 ***
Age18-25                     0.1153657  0.0215789    5.346 8.98e-08 ***
Age26-35                     0.1945145  0.0209692    9.276  < 2e-16 ***
Age36-45                     0.2572174  0.0215513   11.935  < 2e-16 ***
Age46-50                     0.2499432  0.0236833   10.554  < 2e-16 ***
Age51-55                     0.3814411  0.0241499   15.795  < 2e-16 ***
Age55+                       0.3456951  0.0264662   13.062  < 2e-16 ***
Occupation                   0.0023126  0.0005136    4.502 6.72e-06 ***
City_CategoryB               0.0651329  0.0082552    7.890 3.02e-15 ***
City_CategoryC               0.2675380  0.0088579   30.203  < 2e-16 ***
Stay_In_Current_City_Years1  0.0159197  0.0106084    1.501 0.133440
Stay_In_Current_City_Years2  0.0184713  0.0118245    1.562 0.118262
Stay_In_Current_City_Years3 -0.0022555  0.0120262   -0.188 0.851233
Stay_In_Current_City_Years4+ 0.0322954  0.0123173    2.622 0.008743 **
Marital_Status              -0.0245683  0.0071527   -3.435 0.000593 ***
Product_Category_1          -0.1728760  0.0011102 -155.722  < 2e-16 ***
Product_Category_2          -0.0486791  0.0008902  -54.686  < 2e-16 ***
Product_Category_3           0.0073419  0.0014769    4.971 6.65e-07 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 579321  on 430061  degrees of freedom
Residual deviance: 529299  on 430043  degrees of freedom
AIC: 529337

Number of Fisher Scoring iterations: 4
```

h. Use AIC as metric to conclude a better model.

Using AIC as metric: From the above build stepwise forward and backward models, we can see that both the approaches suggest the same model, which is similar to full model.

i. Calculate the Accuracy of the final Logistic Regression model.

```
> model_lg_fit<-train(as.factor(Purchase)~.,data=train.data_lg,method="glm",family="binomial")
> pred<-predict(model_lg_fit,newdata=test.data_lg)
> accuracy(pred,test.data_lg$Purchase)
[1] 0.7606567
> |
```

8. Area Under curve for Best Model:

```
> #Validation
> valid_pred<-knn.1
>
> #Storing Model Performance Scores
> library(ROCR)
> prediction_val<-prediction(as.numeric(knn.1),as.numeric(test.data_knn$Purchase))
>
> #Calculating Area under Curve (AUC)
> performance_value<-performance(perf_val,"auc")
> #Plot AUC
> performance_value<-performance(perf_val,"tpr","fpr")
> plot(performance_value,col="red",lwd=1.5)
>
> AUC<-max(attr(perf_val3,"y.values")[[1]]-(attr(perf_val3,"x.values")[[1]]))
> AUC
[1] 0.9715863
```