Black Friday Sales: Analysis and Prediction

Scenario

A retail store wants to know the customer purchase behavior in terms of purchase amount against various products of different categories. Towards this, store wanted to build a model to predict the purchase amount of each customer against the products they purchased to create personalized offer for their customers.

Objective

The objective this project is build a regression model to predict the dependent variable (the amount of purchase) for the products with the help of the information contained in the other variables

Approach/Activities

The approach includes understanding the customers on the basis of their purchasing habits, according to Age groups, Occupation, City Categories. Customer segmentation/group used to model the data and use to predict the purchase spend for each customer. The activities included: Data exploration, data cleaning, univariate and bivariate analysis, Data Manipulation, One hot-encoding, building different prediction model using h2o for the algorithms multiple regression, random forest, GBM and deep learning. Finally selecting the prediction model with lowest RMSE

Data set information

Source of Dataset: Analytics Vidhya

https://datahack.analyticsvidhya.com/contest/black-friday/

The data set contains customer demographics (age, gender, marital status, city_type, stay_in_current_city), product details (product_id and product category) and Total purchase_amount from last month

Below is the data description:

User_ID: User ID

Product_ID: Product ID

Gender: Sex of User

Age: Age in bins

Occupation: Occupation (Masked)

City_Category: Category of the City (A,B,C)

Stay_In_Current_City_Years: Number of years stay in current city

Marital_Status: Marital Status

Product_Category_1: Product Category (Masked)

Product_Category_2: Product may belongs to other category also (Masked)

Product_Category_3: Product may belongs to other category also (Masked)

Purchase: Purchase Amount (Target Variable)

Initialization

```r
#Library calling and Data loading using fread
library(rpart)

## Warning: package 'rpart' was built under R version 3.5.1

library(data.table)

## Warning: package 'data.table' was built under R version 3.5.1

train <- fread("train_black.csv", stringsAsFactors = T)
test <- fread("test_black.csv", stringsAsFactors = T)

# Data dimension and structure
dim(train)

## [1] 550068     12

dim(test)

## [1] 233599     11

str(train)

## Classes 'data.table' and 'data.frame':   550068 obs. of  12 variables:
##  $ User_ID                   : int  1000001 1000001 1000001 1000001 100000
2 1000003 1000004 1000004 1000004 1000005 ...
##  $ Product_ID                : Factor w/ 3631 levels "P00000142","P0000024
2",..: 673 2377 853 829 2735 1832 1746 3321 3605 2632 ...
##  $ Gender                    : Factor w/ 2 levels "F","M": 1 1 1 1 2 2 2 2
2 2 ...
##  $ Age                       : Factor w/ 7 levels "0-17","18-25",..: 1 1 1
1 7 3 5 5 5 3 ...
##  $ Occupation                : int  10 10 10 10 16 15 7 7 7 20 ...
##  $ City_Category             : Factor w/ 3 levels "A","B","C": 1 1 1 1 3 1
```

```
2 2 2 1 ...
##  $ Stay_In_Current_City_Years: Factor w/ 5 levels "0","1","2","3",..: 3 3 
3 3 5 4 3 3 3 2 ...
##  $ Marital_Status            : int  0 0 0 0 0 0 1 1 1 1 ...
##  $ Product_Category_1        : int  3 1 12 12 8 1 1 1 1 8 ...
##  $ Product_Category_2        : int  NA 6 NA 14 NA 2 8 15 16 NA ...
##  $ Product_Category_3        : int  NA 14 NA NA NA NA 17 NA NA NA ...
##  $ Purchase                  : int  8370 15200 1422 1057 7969 15227 19215 
15854 15686 7871 ...
##  - attr(*, ".internal.selfref")=<externalptr>

#combine data set
test[,Purchase := mean(train$Purchase)]
c <- list(train, test)
combin <- rbindlist(c)
```

Data Exploration of variables using data.table & ggplot-Univariate

```
#Gender

combin[,prop.table(table(Gender))]

## Gender
##         F         M 
## 0.2470896 0.7529104

#Age
combin[,prop.table(table(Age))]

## Age
##       0-17      18-25      26-35      36-45      46-50      51-55 
## 0.02722330 0.18113944 0.39942348 0.19998801 0.08329814 0.06990724 
##        55+ 
## 0.03902040

#City Category
combin[,prop.table(table(City_Category))]

## City_Category
##         A         B         C 
## 0.2682823 0.4207642 0.3109535

#Stay in Current Years
combin[,prop.table(table(Stay_In_Current_City_Years))]

## Stay_In_Current_City_Years
##         0         1         2         3        4+ 
## 0.1348991 0.3527327 0.1855724 0.1728132 0.1539825

#Unique values in Product and User ID
length(unique(combin$Product_ID))

## [1] 3677
```

```r
length(unique(combin$User_ID))
```

```
## [1] 5891
```

```r
#Finding missing values
colSums(is.na(combin))
```

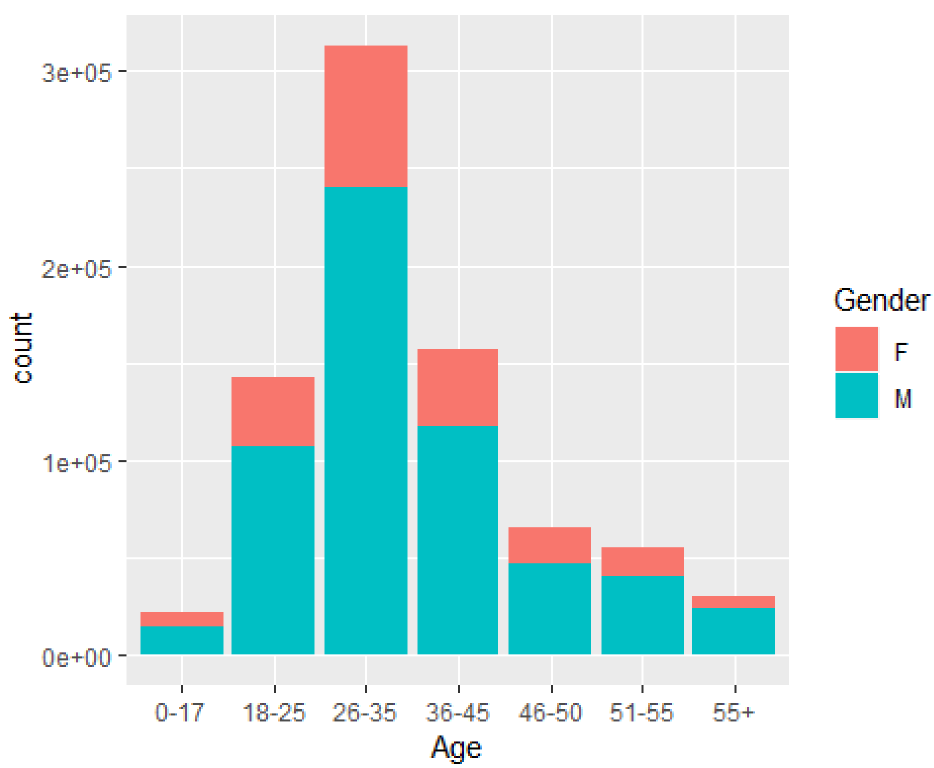
```
##                    User_ID                 Product_ID
##                          0                          0
##                     Gender                        Age
##                          0                          0
##                 Occupation              City_Category
##                          0                          0
## Stay_In_Current_City_Years             Marital_Status
##                          0                          0
##         Product_Category_1         Product_Category_2
##                          0                     245982
##         Product_Category_3                   Purchase
##                     545809                          0
```

Data Exploration of variables using data.table & ggplot-Bivariate

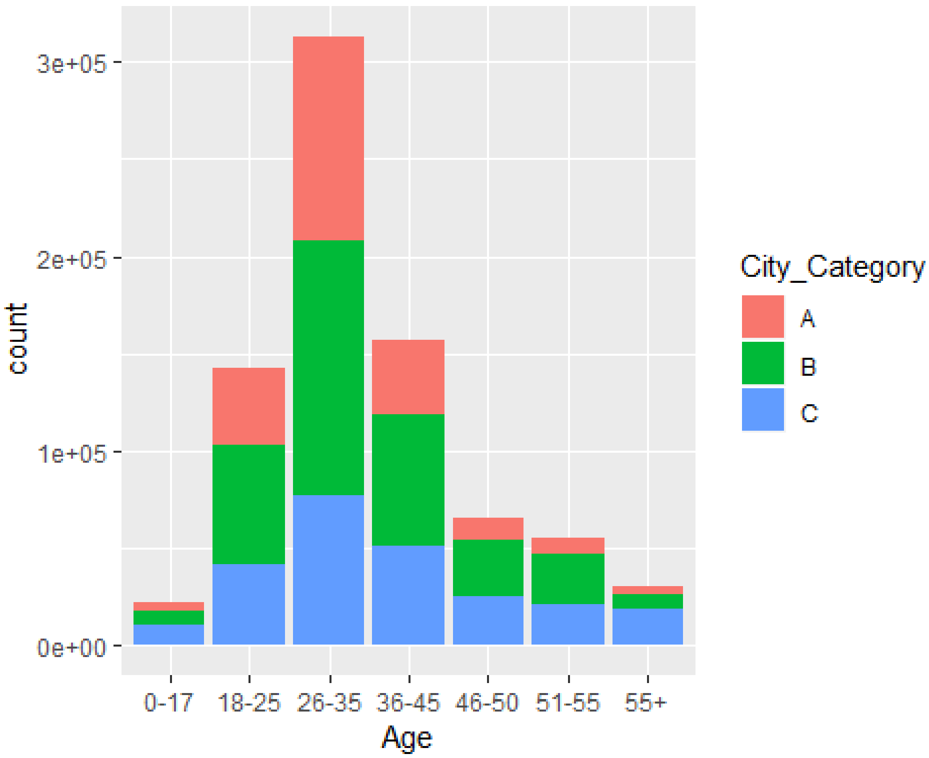
```r
library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 3.5.1
```

```r
#Age vs Gender
ggplot(combin, aes(Age, fill = Gender)) + geom_bar()
```



```r
#Age vs City_Category
ggplot(combin, aes(Age, fill = City_Category)) + geom_bar()
```




```r
#Analyzing categorical variables

library(gmodels)

## Warning: package 'gmodels' was built under R version 3.5.1

CrossTable(combin$Occupation, combin$City_Category)

## 
## 
##    Cell Contents
## |-------------------------|
## |                       N |
## | Chi-square contribution |
## |           N / Row Total |
## |           N / Col Total |
## |         N / Table Total |
## |-------------------------|
## 
## 
## Total Observations in Table:  783667
## 
## 
##                      | combin$City_Category
```

```
## combin$Occupation |         A |         B |         C | Row Total |
## -------------------|-----------|-----------|-----------|-----------|
##                  0 |     26874 |     42455 |     29521 |     98850 |
##                    |     4.733 |    17.884 |    48.165 |           |
##                    |     0.272 |     0.429 |     0.299 |     0.126 |
##                    |     0.128 |     0.129 |     0.121 |           |
##                    |     0.034 |     0.054 |     0.038 |           |
## -------------------|-----------|-----------|-----------|-----------|
##                  1 |     18200 |     28264 |     21223 |     67687 |
##                    |     0.092 |     1.642 |     1.463 |           |
##                    |     0.269 |     0.418 |     0.314 |     0.086 |
##                    |     0.087 |     0.086 |     0.087 |           |
##                    |     0.023 |     0.036 |     0.027 |           |
## -------------------|-----------|-----------|-----------|-----------|
##                  2 |     13201 |     16276 |      8519 |     37996 |
##                    |   887.231 |     5.211 |   919.471 |           |
##                    |     0.347 |     0.428 |     0.224 |     0.048 |
##                    |     0.063 |     0.049 |     0.035 |           |
##                    |     0.017 |     0.021 |     0.011 |           |
## -------------------|-----------|-----------|-----------|-----------|
##                  3 |      8040 |      9747 |      7339 |     25126 |
##                    |   250.378 |    64.398 |    28.759 |           |
##                    |     0.320 |     0.388 |     0.292 |     0.032 |
##                    |     0.038 |     0.030 |     0.030 |           |
##                    |     0.010 |     0.012 |     0.009 |           |
## -------------------|-----------|-----------|-----------|-----------|
##                  4 |     34577 |     42524 |     25985 |    103086 |
##                    |  1731.917 |    16.692 |  1149.411 |           |
##                    |     0.335 |     0.413 |     0.252 |     0.132 |
##                    |     0.164 |     0.129 |     0.107 |           |
##                    |     0.044 |     0.054 |     0.033 |           |
## -------------------|-----------|-----------|-----------|-----------|
##                  5 |      3380 |      9467 |      4526 |     17373 |
##                    |   352.000 |   636.521 |   142.112 |           |
##                    |     0.195 |     0.545 |     0.261 |     0.022 |
##                    |     0.016 |     0.029 |     0.019 |           |
##                    |     0.004 |     0.012 |     0.006 |           |
## -------------------|-----------|-----------|-----------|-----------|
##                  6 |      5321 |     15656 |      8125 |     29102 |
##                    |   791.918 |   950.127 |    94.422 |           |
##                    |     0.183 |     0.538 |     0.279 |     0.037 |
##                    |     0.025 |     0.047 |     0.033 |           |
##                    |     0.007 |     0.020 |     0.010 |           |
## -------------------|-----------|-----------|-----------|-----------|
##                  7 |     22956 |     32859 |     28312 |     84127 |
##                    |     6.609 |   182.064 |   177.101 |           |
##                    |     0.273 |     0.391 |     0.337 |     0.107 |
##                    |     0.109 |     0.100 |     0.116 |           |
##                    |     0.029 |     0.042 |     0.036 |           |
## -------------------|-----------|-----------|-----------|-----------|
```

```
##                   8 |       134 |      1178 |       877 |      2189 |
##                     |   349.845 |    71.681 |    56.624 |           |
##                     |     0.061 |     0.538 |     0.401 |     0.003 |
##                     |     0.001 |     0.004 |     0.004 |           |
##                     |     0.000 |     0.002 |     0.001 |           |
## --------------------|-----------|-----------|-----------|-----------|
##                   9 |       999 |      4574 |      3356 |      8929 |
##                     |   814.109 |   177.664 |   120.949 |           |
##                     |     0.112 |     0.512 |     0.376 |     0.011 |
##                     |     0.005 |     0.014 |     0.014 |           |
##                     |     0.001 |     0.006 |     0.004 |           |
## --------------------|-----------|-----------|-----------|-----------|
##                  10 |      3138 |      6039 |      9127 |     18304 |
##                     |   639.886 |   358.943 |  2073.431 |           |
##                     |     0.171 |     0.330 |     0.499 |     0.023 |
##                     |     0.015 |     0.018 |     0.037 |           |
##                     |     0.004 |     0.008 |     0.012 |           |
## --------------------|-----------|-----------|-----------|-----------|
##                  11 |      3537 |      8002 |      5054 |     16593 |
##                     |   187.912 |   149.093 |     2.163 |           |
##                     |     0.213 |     0.482 |     0.305 |     0.021 |
##                     |     0.017 |     0.024 |     0.021 |           |
##                     |     0.005 |     0.010 |     0.006 |           |
## --------------------|-----------|-----------|-----------|-----------|
##                  12 |     10057 |     18784 |     15607 |     44448 |
##                     |   292.502 |     0.358 |   230.722 |           |
##                     |     0.226 |     0.423 |     0.351 |     0.057 |
##                     |     0.048 |     0.057 |     0.064 |           |
##                     |     0.013 |     0.024 |     0.020 |           |
## --------------------|-----------|-----------|-----------|-----------|
##                  13 |       561 |      3466 |      7026 |     11053 |
##                     |  1949.458 |   301.788 |  3747.820 |           |
##                     |     0.051 |     0.314 |     0.636 |     0.014 |
##                     |     0.003 |     0.011 |     0.029 |           |
##                     |     0.001 |     0.004 |     0.009 |           |
## --------------------|-----------|-----------|-----------|-----------|
##                  14 |     10975 |     15971 |     11836 |     38782 |
##                     |    31.279 |     7.382 |     4.138 |           |
##                     |     0.283 |     0.412 |     0.305 |     0.049 |
##                     |     0.052 |     0.048 |     0.049 |           |
##                     |     0.014 |     0.020 |     0.015 |           |
## --------------------|-----------|-----------|-----------|-----------|
##                  15 |      4373 |      7479 |      5504 |     17356 |
##                     |    17.238 |     4.252 |     2.125 |           |
##                     |     0.252 |     0.431 |     0.317 |     0.022 |
##                     |     0.021 |     0.023 |     0.023 |           |
##                     |     0.006 |     0.010 |     0.007 |           |
## --------------------|-----------|-----------|-----------|-----------|
##                  16 |      8772 |     15444 |     11906 |     36122 |
##                     |    87.130 |     3.954 |    40.412 |           |
```

```
##                    |      0.243 |      0.428 |      0.330 |      0.046 |
##                    |      0.042 |      0.047 |      0.049 |            |
##                    |      0.011 |      0.020 |      0.015 |            |
## -------------------|------------|------------|------------|------------|
##                 17 |      11668 |      23204 |      22546 |      57418 |
##                    |    906.208 |     37.785 |   1232.854 |            |
##                    |      0.203 |      0.404 |      0.393 |      0.073 |
##                    |      0.055 |      0.070 |      0.093 |            |
##                    |      0.015 |      0.030 |      0.029 |            |
## -------------------|------------|------------|------------|------------|
##                 18 |       2246 |       3030 |       4091 |       9367 |
##                    |     28.368 |    210.708 |    476.667 |            |
##                    |      0.240 |      0.323 |      0.437 |      0.012 |
##                    |      0.011 |      0.009 |      0.017 |            |
##                    |      0.003 |      0.004 |      0.005 |            |
## -------------------|------------|------------|------------|------------|
##                 19 |       3165 |       4712 |       4042 |      11919 |
##                    |      0.334 |     18.317 |     30.415 |            |
##                    |      0.266 |      0.395 |      0.339 |      0.015 |
##                    |      0.015 |      0.014 |      0.017 |            |
##                    |      0.004 |      0.006 |      0.005 |            |
## -------------------|------------|------------|------------|------------|
##                 20 |      18070 |      20608 |       9162 |      47840 |
##                    |   2135.562 |     11.381 |   2194.806 |            |
##                    |      0.378 |      0.431 |      0.192 |      0.061 |
##                    |      0.086 |      0.062 |      0.038 |            |
##                    |      0.023 |      0.026 |      0.012 |            |
## -------------------|------------|------------|------------|------------|
##       Column Total |     210244 |     329739 |     243684 |     783667 |
##                    |      0.268 |      0.421 |      0.311 |            |
## -------------------|------------|------------|------------|------------|
##
##
```

Data Manipulation using data.table

```
#Creating new variables,revalue existing variable and treat missing values
#Missing value treatment for Product_Category_2 and Product_Category_3
combin[,Product_Category_2_NA := ifelse(sapply(combin$Product_Category_2, is.
na) ==    TRUE,1,0)]
combin[,Product_Category_3_NA := ifelse(sapply(combin$Product_Category_3, is.
na) ==  TRUE,1,0)]

#Impute missing values
combin[,Product_Category_2 := ifelse(is.na(Product_Category_2) == TRUE, "-999
",  Product_Category_2)]
combin[,Product_Category_3 := ifelse(is.na(Product_Category_3) == TRUE, "-999
",  Product_Category_3)]

#Revaluing Stay_In_Current_City_Years variable levels
```

```r
levels(combin$Stay_In_Current_City_Years)[levels(combin$Stay_In_Current_City_
Years) ==  "4+"] <- "4"

#Re-coding age groups
levels(combin$Age)[levels(combin$Age) == "0-17"] <- 0
levels(combin$Age)[levels(combin$Age) == "18-25"] <- 1
levels(combin$Age)[levels(combin$Age) == "26-35"] <- 2
levels(combin$Age)[levels(combin$Age) == "36-45"] <- 3
levels(combin$Age)[levels(combin$Age) == "46-50"] <- 4
levels(combin$Age)[levels(combin$Age) == "51-55"] <- 5
levels(combin$Age)[levels(combin$Age) == "55+"] <- 6

#convert age to numeric
combin$Age <- as.numeric(combin$Age)

#convert Gender into numeric
combin$Gender<- as.numeric(combin$Gender)

#New variable to capture count of ID variables
combin[, User_Count := .N, by = User_ID]

combin[, Product_Count := .N, by = Product_ID]

#Mean purchase of user and prodcut

combin[, Mean_Purchase_Product := mean(Purchase), by = Product_ID]

combin[, Mean_Purchase_User := mean(Purchase), by = User_ID]

#One-hot encoding of variable City_Category

library(dummies)

## dummies-1.5.6 provided by Decision Patterns

combin <- dummy.data.frame(combin, names = c("City_Category"), sep = "_")

#checking classes of all variables
sapply(combin, class)

##                    User_ID                 Product_ID
##                  "integer"                   "factor"
##                     Gender                        Age
##                  "numeric"                  "numeric"
##                 Occupation            City_Category_A
##                  "integer"                  "integer"
##            City_Category_B            City_Category_C
##                  "integer"                  "integer"
## Stay_In_Current_City_Years             Marital_Status
##                   "factor"                  "integer"
##         Product_Category_1         Product_Category_2
```

```
##                  "integer"                   "character"
##         Product_Category_3                      Purchase
##                "character"                     "numeric"
##      Product_Category_2_NA      Product_Category_3_NA
##                  "numeric"                     "numeric"
##                 User_Count                 Product_Count
##                  "integer"                     "integer"
##      Mean_Purchase_Product         Mean_Purchase_User
##                  "numeric"                     "numeric"

#converting Product Category 2 & 3 to integer
combin$Product_Category_2 <- as.integer(combin$Product_Category_2)
combin$Product_Category_3 <- as.integer(combin$Product_Category_3)
```

Model Building using H2O

```
##Dividing into train and test
c.train <- combin[1:nrow(train),]
c.test <- combin[-(1:nrow(train)),]

#Dropping rows which has category level 19 & 20 in Product_Category_1

c.train <- c.train[c.train$Product_Category_1 <= 18,]

#Initiating h2o

library(h2o)

localH2O <- h2o.init(nthreads = -1)

h2o.init()

##  Connection successful!
##
## R is connected to the H2O cluster:
##     H2O cluster uptime:         1 days 44 minutes
##     H2O cluster timezone:       Asia/Kolkata
##     H2O data parsing timezone:  UTC
##     H2O cluster version:        3.20.0.8
##     H2O cluster version age:    3 months and 5 days
##     H2O cluster name:           H2O_started_from_R_186481_fmx062
##     H2O cluster total nodes:    1
##     H2O cluster total memory:   1.82 GB
##     H2O cluster total cores:    4
##     H2O cluster allowed cores:  4
##     H2O cluster healthy:        TRUE
##     H2O Connection ip:          localhost
##     H2O Connection port:        54321
##     H2O Connection proxy:       NA
##     H2O Internal Security:      FALSE
```

```
##     H2O API Extensions:         Algos, AutoML, Core V3, Core V4
##     R Version:                  R version 3.5.0 (2018-04-23)
```

```r
#Transfering data from R to h2o instancer

train.h2o <- as.h2o(c.train)

#checking column index number

colnames(train.h2o)
```

```
##  [1] "User_ID"                    "Product_ID"
##  [3] "Gender"                     "Age"
##  [5] "Occupation"                 "City_Category_A"
##  [7] "City_Category_B"            "City_Category_C"
##  [9] "Stay_In_Current_City_Years" "Marital_Status"
## [11] "Product_Category_1"         "Product_Category_2"
## [13] "Product_Category_3"         "Purchase"
## [15] "Product_Category_2_NA"      "Product_Category_3_NA"
## [17] "User_Count"                 "Product_Count"
## [19] "Mean_Purchase_Product"      "Mean_Purchase_User"
```

```r
#Dependent variable (Purchase)
y.dep <- 14

#Independent variables (dropping ID variables)
x.indep <- c(3:13,15:20)
```

Multiple Regression in H2O

```r
regression.model <- h2o.glm( y = y.dep, x = x.indep, training_frame = train.h
2o, family = "gaussian")


  h2o.performance(regression.model)
```

```
## H2ORegressionMetrics: glm
## ** Reported on training data. **
##
## MSE:  16710563
## RMSE:  4087.856
## MAE:  3219.644
## RMSLE:  0.5782911
## Mean Residual Deviance :  16710563
## R^2 :  0.3261543
## Null Deviance :1.353804e+13
## Null D.o.F. :545914
## Residual Deviance :9.122547e+12
## Residual D.o.F. :545898
## AIC :10628689
```

Random Forest in H2O

```
rforest.model <- h2o.randomForest(y=y.dep, x=x.indep, training_frame = train.
h2o, ntrees = 1000, mtries = 3, max_depth = 4, seed = 1122)

h2o.performance(rforest.model)
```

```
## H2ORegressionMetrics: drf
## ** Reported on training data. **
## ** Metrics reported on Out-Of-Bag training samples **
##
## MSE:  10414919
## RMSE:  3227.215
## MAE:  2486.118
## RMSLE:  0.5007453
## Mean Residual Deviance :  10414919
```

```
#check variable importance
h2o.varimp(rforest.model)
```

```
## Variable Importances:
##                      variable     relative_importance scaled_importance
## 1        Mean_Purchase_Product 2720452686381056.000000          1.000000
## 2           Product_Category_1 1005997304840192.000000          0.369790
## 3                Product_Count  252741091852288.000000          0.092904
## 4           Product_Category_3  231408274505728.000000          0.085062
## 5        Product_Category_3_NA  194243133964288.000000          0.071401
## 6           Mean_Purchase_User  174858721820672.000000          0.064276
## 7           Product_Category_2   84932466573312.000000          0.031220
## 8        Product_Category_2_NA   54471002423296.000000          0.020023
## 9                   User_Count   12314694647808.000000          0.004527
## 10             City_Category_C    5007590031360.000000          0.001841
## 11                      Gender    2175469223936.000000          0.000800
## 12             City_Category_A    1162100736000.000000          0.000427
## 13                         Age     613729370112.000000          0.000226
## 14                  Occupation     478127718400.000000          0.000176
## 15             City_Category_B     234770481152.000000          0.000086
## 16 Stay_In_Current_City_Years      32139771904.000000          0.000012
## 17              Marital_Status      17185155072.000000          0.000006
##    percentage
## 1    0.573797
## 2    0.212185
## 3    0.053308
## 4    0.048809
## 5    0.040970
## 6    0.036881
## 7    0.017914
## 8    0.011489
## 9    0.002597
## 10   0.001056
## 11   0.000459
```

```
## 12   0.000245
## 13   0.000129
## 14   0.000101
## 15   0.000050
## 16   0.000007
## 17   0.000004
```

Gradient Boosting Machine in H2O

```
gbm.model <- h2o.gbm(y=y.dep, x=x.indep, training_frame = train.h2o, ntrees =
1000, max_depth = 4, learn_rate = 0.01, seed = 1122)
```

```
h2o.performance (gbm.model)
```

```
## H2ORegressionMetrics: gbm
## ** Reported on training data. **
##
## MSE:  6321280
## RMSE:  2514.216
## MAE:  1859.895
## RMSLE:  NaN
## Mean Residual Deviance :  6321280
```

Deep Learning in H2O

```
dlearning.model <- h2o.deeplearning(y = y.dep,
                                    x = x.indep,
                                    training_frame = train.h2o,
                                    epoch = 60,
                                    hidden = c(100,100),
                                    activation = "Rectifier",
                                    seed = 1122)
```

```
h2o.performance(dlearning.model)
```

```
## H2ORegressionMetrics: deeplearning
## ** Reported on training data. **
## ** Metrics reported on temporary training frame with 9881 samples **
##
## MSE:  6163649
## RMSE:  2482.67
## MAE:  1825.37
## RMSLE:  NaN
## Mean Residual Deviance :  6163649
```

```
dlearning.model
```

```
## Model Details:
## ==============
##
## H2ORegressionModel: deeplearning
## Model ID:  DeepLearning_model_R_1545817681819_18
## Status of Neuron Layers: predicting Purchase, regression, gaussian distrib
```

```
ution, Quadratic loss, 12,501 weights/biases, 154.4 KB, 13,097,784 training s
amples, mini-batch size 1
##   layer units      type dropout       l1       l2 mean_rate rate_rms
## 1     1    22     Input  0.00 %       NA       NA        NA       NA
## 2     2   100 Rectifier  0.00 % 0.000000 0.000000  0.050958 0.207950
## 3     3   100 Rectifier  0.00 % 0.000000 0.000000  0.040882 0.051702
## 4     4     1    Linear      NA 0.000000 0.000000  0.000982 0.001225
##   momentum mean_weight weight_rms mean_bias bias_rms
## 1       NA          NA         NA        NA       NA
## 2 0.000000   -0.075226   0.548449 -0.814823 0.466772
## 3 0.000000   -0.114927   0.242933 -0.361529 1.011846
## 4 0.000000    0.019601   0.107993  0.274263 0.000000
##
##
## H2ORegressionMetrics: deeplearning
## ** Reported on training data. **
## ** Metrics reported on temporary training frame with 9881 samples **
##
## MSE:  6163649
## RMSE:  2482.67
## MAE:  1825.37
## RMSLE:  NaN
## Mean Residual Deviance :  6163649
```

```r
#From above algorithms, deeplearning has lowest RMSE value
##Making predictions based on deeplearning
predict.dl2 <- as.data.frame(h2o.predict(dlearning.model, test.h2o))

#creating a data frame and writing csv file for predicted values
sub_dlearning <- data.frame(User_ID = test$User_ID, Product_ID = test$Product
_ID, Purchase = predict.dl2$predict)
write.csv(sub_dlearning, file = "sub_dlearning_new.csv", row.names = F)
```