R Notebook

```
#setwd("~/R_KSU/ML/Assignment 3")
bank <- read.csv("UniversalBank.csv")
```

Importing libraries

```
library(dplyr)
library(reshape)
library(reshape2)
library(ggplot2)
library(caret)
```

Exploratory Data Analysis

```
str(bank)
```

```
## 'data.frame':    5000 obs. of  14 variables:
##  $ ID                : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ Age               : int  25 45 39 35 35 37 53 50 35 34 ...
##  $ Experience        : int  1 19 15 9 8 13 27 24 10 9 ...
##  $ Income            : int  49 34 11 100 45 29 72 22 81 180 ...
##  $ ZIP.Code          : int  91107 90089 94720 94112 91330 92121 91711 93943 90089 93023 ...
##  $ Family            : int  4 3 1 1 4 4 2 1 3 1 ...
##  $ CCAvg             : num  1.6 1.5 1 2.7 1 0.4 1.5 0.3 0.6 8.9 ...
##  $ Education         : int  1 1 1 2 2 2 2 3 2 3 ...
##  $ Mortgage          : int  0 0 0 0 0 155 0 0 104 0 ...
##  $ Personal.Loan     : int  0 0 0 0 0 0 0 0 0 1 ...
##  $ Securities.Account: int  1 1 0 0 0 0 0 0 0 0 ...
##  $ CD.Account        : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ Online            : int  0 0 0 0 0 1 1 0 1 0 ...
##  $ CreditCard        : int  0 0 0 0 1 0 0 1 0 0 ...
```

> From the above data table stucture we found that - ‘Online’, ‘CreditCard’ & ‘Personal.Loan’ columns have integer data type. But, these are nominal data and we need to convert them to categorical data.

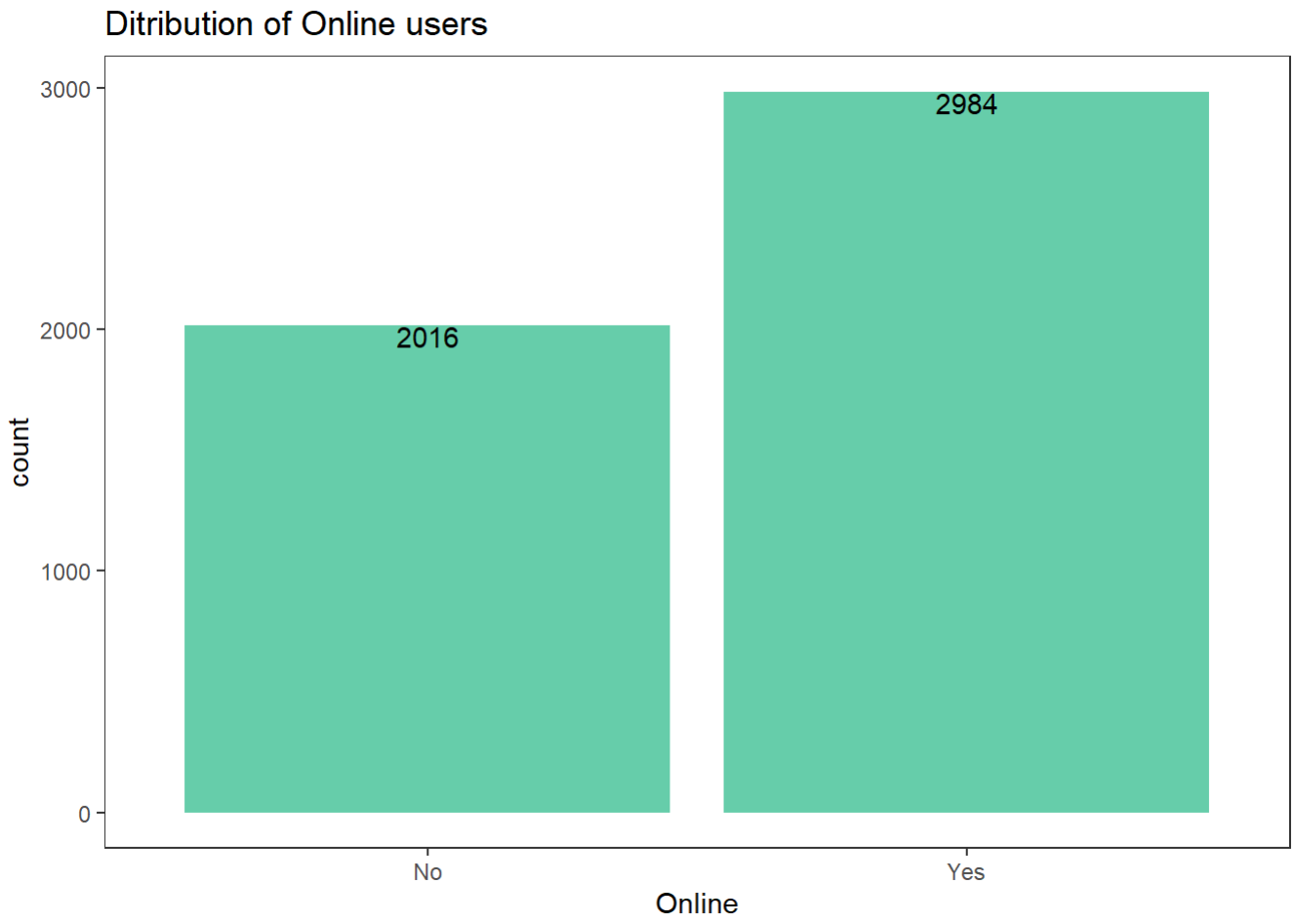
```
bank$Personal.Loan = as.factor(bank$Personal.Loan)
bank$Online = as.factor(bank$Online)
bank$CreditCard = as.factor(bank$CreditCard)
```

Summary statistics of categorical variables.

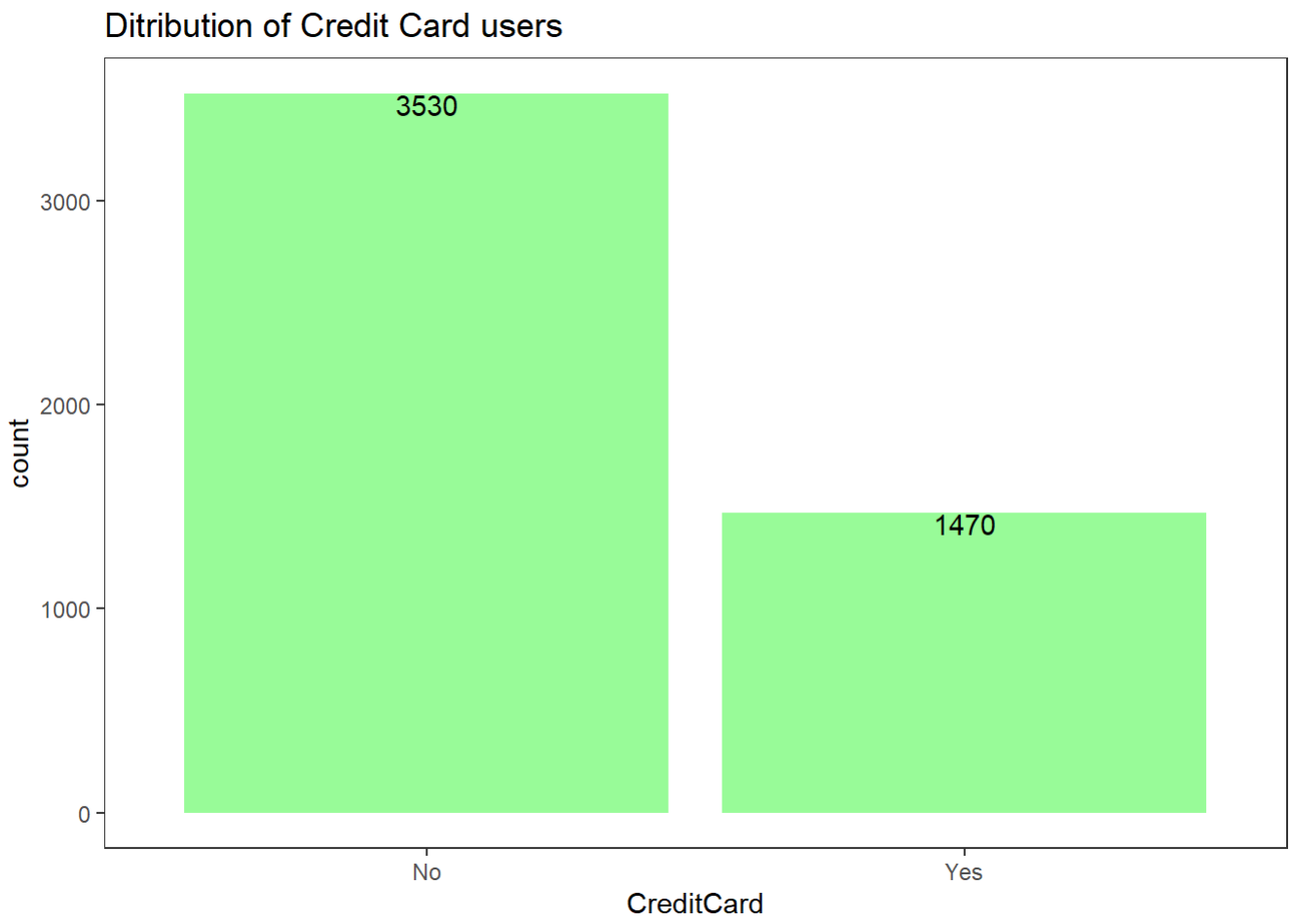
```
summary(bank %>% select(Online, CreditCard, Personal.Loan))
```

```
##  Online   CreditCard Personal.Loan
##  0:2016   0:3530     0:4520
##  1:2984   1:1470     1: 480
```

```
ggplot(bank, aes(x=Online)) + geom_bar( fill='mediumaquamarine') + ggtitle('Ditribution of Onlin
e users') + theme_test() + scale_x_discrete(labels=c("No", "Yes")) + geom_text(aes(label=..coun
t..),stat="count", vjust=1)
```

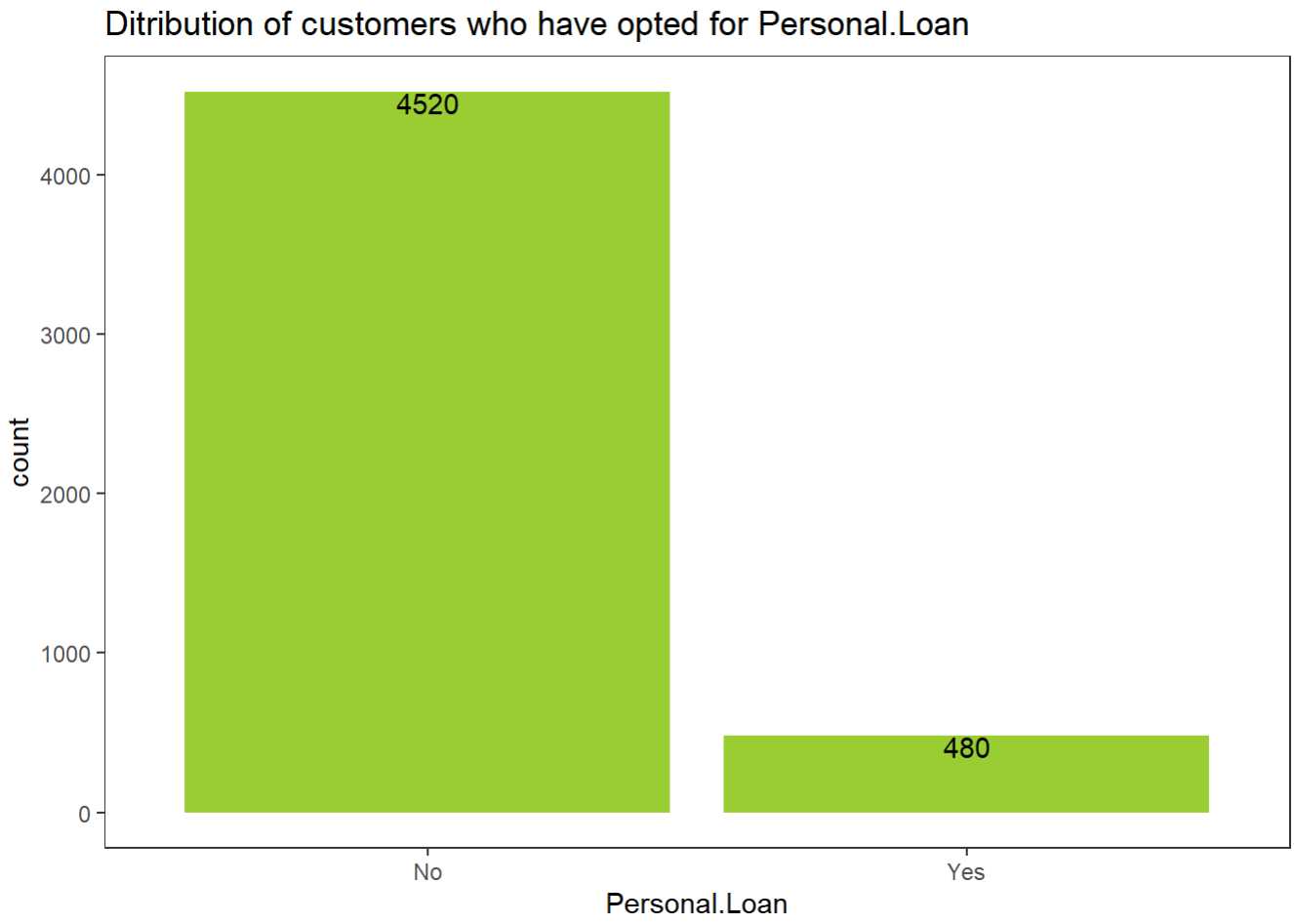



```
ggplot(bank, aes(x=CreditCard)) + geom_bar( fill='palegreen') + ggtitle('Ditribution of Credit C
ard users') + theme_test() + scale_x_discrete(labels=c("No", "Yes")) + geom_text(aes(label=..cou
nt..),stat="count", vjust=1)
```




```
ggplot(bank, aes(x=Personal.Loan)) + geom_bar( fill='olivedrab3') + ggtitle('Ditribution of cust
omers who have opted for Personal.Loan') + theme_test() + scale_x_discrete(labels=c("No", "Yes"
)) + scale_x_discrete(labels=c("No", "Yes")) + geom_text(aes(label=..count..),stat="count", vjus
t=1)
```

```
## Scale for 'x' is already present. Adding another scale for 'x', which will
## replace the existing scale.
```



Data Preprocessing

Splitting data into train & test (60-40).

```
set.seed(1)
train.index <- createDataPartition(bank$Personal.Loan, p=0.6, list= FALSE)
train <- bank[train.index,]
test <- bank[-train.index,]
```

A. Create a pivot table for the training data with Online as a column variable, CC as a row variable, and Loan as a secondary row variable. The values inside the table should convey the count. In R use functions melt() and cast(), or function table(). In Python, use panda dataframe methods melt() and pivot().

```
t1= melt(train,id=c('CreditCard','Personal.Loan'),variable='Online')
t1= dcast(t1, CreditCard+Personal.Loan~Online)
t1
```

CreditCard <fct>	Personal.Loan <fct>	ID <int>	A… <int>	Experience <int>	Inco… <int>	ZIP.Code <int>	Fam… <int>	CC… <int>	Education <int>
0	0	1906	1906	1906	1906	1906	1906	1906	1906
0	1	197	197	197	197	197	197	197	197
1	0	806	806	806	806	806	806	806	806
1	1	91	91	91	91	91	91	91	91

4 rows | 1-10 of 14 columns

B. Consider the task of classifying a customer who owns a bank credit card and is actively using online banking services. Looking at the pivot table, what is the probability that this customer will accept the loan offer? [This is the probability of loan acceptance (Loan = 1) conditional on having a bank credit card (CC = 1) and being an active user of online banking services (Online = 1)].

Probability of Loan acceptance given having a bank credit card and user of online services is 91/91+806 = 0.10144 => [10.144%]

C. Create two separate pivot tables for the training data. One will have Loan (rows) as a function of Online (columns) and the other will have Loan (rows) as a function of CC.

```
as.data.frame(table(train[,c('Personal.Loan','CreditCard')]))
```

Personal.Loan <fct>	CreditCard <fct>	Freq <int>
0	0	1906
1	0	197
0	1	806
1	1	91

4 rows

```
as.data.frame(table(train[,c('Personal.Loan','Online')]))
```

Personal.Loan <fct>	Online <fct>	Freq <int>
0	0	1083
1	0	116
0	1	1629
1	1	172

4 rows

D. Compute the following quantities [P(A | B) means “the probability of A given B”]: i. P(CC = 1 | Loan = 1) (the proportion of credit card holders among the loan acceptors) ii. P(Online = 1 | Loan = 1) iii. P(Loan = 1) (the proportion of loan acceptors) iv. P(CC = 1 | Loan = 0) v. P(Online = 1 | Loan = 0) vi. P(Loan = 0)

```
as.data.frame(table(train[,c(14,10)]))
```

CreditCard <fct>	Personal.Loan <fct>	Freq <int>
0	0	1906
1	0	806
0	1	197
1	1	91

4 rows

```
as.data.frame(table(train[,c(13,10)]))
```

Online <fct>	Personal.Loan <fct>	Freq <int>
0	0	1083
1	0	1629
0	1	116
1	1	172

4 rows

```
as.data.frame(table(train[,c(10)]))
```

Var1 <fct>	Freq <int>
0	2712
1	288

2 rows

P(Cc|Pl) = 91/(91+197) = 0.31597 P(Ol|Pl) = 172/(172+116) = 0.59722 P(Pl) = 288/(288+2712) = 0.09600 P(Cc|Pl')= 806/(806+1906) = 0.29719 P(Ol|Pl')= 1629/(1629+1083)= 0.60066 P(Pl') = 2712/(2712+288) = 0.90400

E. Use the quantities computed above to compute the naive Bayes probability P(Loan = 1 | CC = 1, Online = 1).

```
(0.31597*0.59722*0.09600) / ((0.31597*0.59722*0.09600) + (0.29719*0.60066*0.90400))
```

```
## [1] 0.1009286
```

F. Compare this value with the one obtained from the pivot table in (B). Which is a more accurate estimate?

10.092% are very similar to the 10.144% the difference between the exact method and the naive-bayes method is the exact method would need the the exact same independent variable classifications to predict, where the naive bayes method does not.

G. Which of the entries in this table are needed for computing P(Loan = 1 | CC = 1, Online = 1)? Run naive Bayes on the data. Examine the model output on training data, and find the entry that corresponds to P(Loan = 1 | CC = 1, Online = 1). Compare this to the number you obtained in (E).

```
library('e1071')
train = train[,c(10,13:14)]
test = test[,c(10,13:14)]
naivebayes = naiveBayes(Personal.Loan~.,data=train)
naivebayes
```

```
##
## Naive Bayes Classifier for Discrete Predictors
##
## Call:
## naiveBayes.default(x = X, y = Y, laplace = laplace)
##
## A-priori probabilities:
## Y
##     0     1
## 0.904 0.096
##
## Conditional probabilities:
##    Online
## Y           0         1
##   0 0.3993363 0.6006637
##   1 0.4027778 0.5972222
##
##    CreditCard
## Y           0         1
##   0 0.7028024 0.2971976
##   1 0.6840278 0.3159722
```

```
(0.3159)*(0.5972)*(0.096)/((0.3159)*(0.5972)*(0.096) + (0.2971)*(0.6006)*(0.904))
```

```
## [1] 0.100942
```

The naive bayes is the exact same output we retrieved in the previous methods. (0.3159)*(0.5972)*(0.096)/((0.3159)*(0.5972)*(0.096) + (0.2971)*(0.6006)*(0.904) = 0.100942 which is almost the same response provided as above.