پذیرش وام شخصی

یونیورسال بانک یک بانک نسبتاً جوان است که از نظر جذب مشتری به سرعت در حال رشد است.

اکثر این مشتریان، مشتریان بدهی (سپرده گذاران) با اندازه های مختلف در ارتباط با بانک هستند.

پایگاه مشتریان دارایی (وام گیرندگان) بسیار کوچک است، و بانک علاقه مند است که این پایگاه را به سرعت گسترش دهد تا تجارت وام بیشتری ایجاد کند.

به ویژه، می‌خواهد راه‌هایی را برای تبدیل مشتریان تعهدی خود به مشتریان وام شخصی (در حالی که آنها را به عنوان سپرده‌گذار حفظ می‌کند) بررسی کند.

کمپینی که بانک در سال گذشته برای مشتریان بدهی اجرا کرد، نرخ تبدیل بالای ۹ درصد را نشان داد.

این امر بخش بازاریابی خرده‌فروشی را تشویق می‌کند تا کمپین‌های هوشمندتر با بازاریابی هدفمند بهتر طراحی کند.

هدف استفاده از **k-NN** برای پیش بینی این است که آیا مشتری جدید پیشنهاد وام را می پذیرد یا خیر.

این به عنوان پایه ای برای طراحی یک کمپین جدید عمل می کند.

فایل **UniversalBank.csv** حاوی اطلاعات ۵۰۰۰ مشتری است.

داده ها شامل اطلاعات جمعیت شناختی مشتری (سن، درآمد و غیره)، رابطه مشتری با بانک (رهن، حساب اوراق بهادار و غیره) و پاسخ مشتری به آخرین کمپین وام شخصی (وام شخصی) است.

از بین این ۵۰۰۰ مشتری، تنها ۴۸۰ نفر (= ۹.۶٪) وام شخصی را که در کمپین قبلی به آنها پیشنهاد شده بود، پذیرفتند.

داده ها را به مجموعه های آموزشی (۶۰٪) و اعتبارسنجی (۴۰٪) تقسیم کنید.

۱. مشتری زیر را در نظر بگیرید:

Age = 40 , Experience = 10 , Income = 84 , Family = 2 , CCAvg = 2 , Education_1= 0 , Education_2 = 1 , Education_3=0 , Mortgage = 0 , Securities Account = 0 , CD Account = 0 , Online = 1 and Credit Card = 1

یک طبقه‌بندی **k-NN** با همه پیش‌بینی‌کننده‌ها به جز **ID** و کد پستی با استفاده از **k = 1** انجام دهید.

به یاد داشته باشید که ابتدا پیش بینی کننده های طبقه بندی شده با بیش از دو دسته را به متغیرهای ساختگی تبدیل کنید.

کلاس موفقیت را به عنوان ۱ (پذیرش وام) مشخص کنید و از مقدار قطع پیش فرض ۰.۵ استفاده کنید. این مشتری چگونه طبقه بندی می شود؟

۲. انتخاب **k** که بین تطبیق بیش از حد و نادیده گرفتن اطلاعات پیش بینی کننده تعادل برقرار می کند چیست؟

۳. ماتریس سردرگمی را برای داده های اعتبارسنجی که از بهترین **k** حاصل می شود نشان دهید.

۴. مشتری زیر را در نظر بگیرید:

Age = 40, Experience = 10, Income = 84, Family = 2, CCAvg = 2, Education_1 = 0, Education_2 = 1, Education_3 = 0, Mortgage = 0, Securities Account = 0, CD Account = 0, Online = 1 and Credit Card = 1

با استفاده از بهترین **K** طبقه بندی کنید.

۵. داده ها را مجدداً تقسیم کنید، این بار به مجموعه های آموزشی، اعتبار سنجی و آزمایش (۵۰٪ : ۳۰٪ :۲۰٪).

روش **k-NN** را با **k** انتخاب شده در بالا اعمال کنید. ماتریس سردرگمی مجموعه آزمون را با مجموعه های آموزشی و اعتبارسنجی مقایسه کنید. در مورد تفاوت ها و دلیل آنها نظر دهید.

ابتدا داده‌های خود را فراخوانی میکنیم

```
#Read Data
> Customers <-read.csv("~/UniversalBank.csv",header = T)
> str(Customers)
'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 930
 $ 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 ...
```

برای ۵۰۰۰ مشتری ۱۴ متغیر اندازه گیری شده است .

متغیر اول یعنی **ID** و **ZIP.Code** را از متغیرهای خود حذف میکنیم.

```
# Delete columns ID and Zip
```

```
> Customers.df<-Customers[,c(-1,-5)]
#Remaining variables
> names(Customers.df)
 [1] "Age"               "Experience"         "Income"             "Family"
"CCAvg"              "Education"
 [7] "Mortgage"          "Personal.Loan"      "Securities.Account" "CD.Account"
"Online"             "CreditCard"
```

لیست متغیرهای باقی‌مانده به صورت بالا است.

بعضی از متغیرهای ما از جنس طبقه‌بندی هستند که باید آنها را با دستور زیر **factor** کنیم :

```
> # as.factor
> Customers.df$Personal.Loan<-as.factor(Customers.df$Personal.Loan)
> Customers.df$Education<-as.factor(Customers.df$Education)
> Customers.df$Family<-as.factor(Customers.df$Family)
> Customers.df$Securities.Account<-as.factor(Customers.df$Securities.Account)
> Customers.df$CD.Account<-as.factor(Customers.df$CD.Account)
> Customers.df$CreditCard<-as.factor(Customers.df$CreditCard)
> Customers.df$Online<-as.factor(Customers.df$Online)
```

۶۰٪ داده‌ها را در **Train** و **40%** داده‌ها را در **Valid** قرار میدهیم :

```
> # 60/40 split for training and validation
> library(caret)
> trainIndex <- createDataPartition(Customers.df$Personal.Loan, p=.6,list = FALSE, ti
mes = 1)
> train.df <- Customers.df[trainIndex,]
> valid.df <- Customers.df[-trainIndex,]
```

چون داده‌های ما دارای مقیاس‌های مختلفی هستند پس نیاز داریم آنها را نرمال کنیم

```
> # use preProcess() from the caret package to normalize Data.
> norm.values <- preProcess(train.df, method=c("center", "scale"))
> train.norm.df <- as.data.frame(predict(norm.values, train.df))
> valid.norm.df <- as.data.frame(predict(norm.values, valid.df))
> Customers.norm.df <- as.data.frame(predict(norm.values, Customers.df))
```

در قسمت الف این سوال از ما میخواهد که برای یک مشتری با متغیرهای داده شده پیش بینی کنیم که آیا شخص وام را می‌پذیر یا خیر.

ما اول متغیرهای داده‌شده را نرمال میکنیم سپس در مدل قرار میدهیم

```
> new.df<-data.frame(Age = 40 , Experience = 10 ,Income = 84 ,Family = 2 ,CCAvg = 2,E
ducation = 2,Mortgage = 0 ,Securities.Account = 0
+                   ,CD.Account = 0,Online = 1,CreditCard = 1)
>
> #norm your new data
```

```
> new.norm.values <- preProcess(new.df, method=c("center", "scale"))
>
> new.norm.df <- predict(new.norm.values, newdata = new.df)
>
> preds.k.1 <- class::knn (train=train.norm.df[,-8], test=new.norm.df, cl=train.df$Personal.Loan, k=1, prob=TRUE)
> head(preds.k.1)
[1] 0
Levels: 0 1
```

مدل ما پیش‌بینی میکند که فرد مورد نظر با متغیرهای داده‌شده وام را نمیپذیرد.

در این قسمت ما بهترین K را برای K=1:20 به کمک confusionMatrix بدست می‌آوریم :

```
> # Accuracy.df
> # initialize a data frame with two columns: k, and accuracy.
> accuracy.df <- data.frame(k = seq(1, 20, 1), accuracy = rep(0, 20))
> # compute knn for different k on validation.
> for(i in 1:20) {
+   knn.pred<-class::knn (train=train.norm.df[,-8], test=valid.norm.df[,-8], cl=train.df$Personal.Loan, k=i, prob=TRUE)
+   accuracy.df[i, 2] <- confusionMatrix(knn.pred, valid.df$Personal.Loan)$overall[1]
+ }
>
> accuracy.df
    k accuracy
1   1   0.9630
2   2   0.9595
3   3   0.9690
4   4   0.9645
5   5   0.9670
6   6   0.9610
7   7   0.9620
8   8   0.9610
9   9   0.9610
10 10   0.9605
11 11   0.9575
12 12   0.9560
13 13   0.9560
14 14   0.9560
15 15   0.9540
16 16   0.9535
17 17   0.9540
18 18   0.9540
19 19   0.9530
20 20   0.9510
> which.max(accuracy.df$accuracy)
[1] 3
```

مدل ما با K=3 بهترین و بالاترین دقت را دارد ، یعنی بهتر از همه پیشبینی میکند.

البته اگر دقت کنید میبینند برای K=1:10 دقت مدل ما زیاد تفات ندارند .

اکنون برای مشتری مورد نظر با K=3 پیش بینی انجام میدهیم :

```
> preds.k.3 <- class::knn (train=train.norm.df[,-8], test=new.norm.df, cl=train.df$Pe
rsonal.Loan, k=3, prob=TRUE)
> head(preds.k.3)
[1] 0
Levels: 0 1
```

با k=3 نیز مدل ما میگوید فرد مورد نظر وام را نمیپذیرد.

در این قسمت سوال از ما میخواهد داده‌های خود را بصورت زیر تقسیم کنیم :

Train = 50% Valid=30% test=20%

برای این کار ما اول ۵۰٪ از داده‌ها را خارج میکنیم و در مجموعه **Train** قرار میدهیم .

در مرحله بعد از ۵۰٪ داده باقی مانده ۶۰٪ داده را استخراج میکنیم که در واقع میشود ۳۰٪ از داده‌های کل.

حال ۲۰٪ از داده‌های کل باقی می‌ماند که انها را در **Test** میریزیم.

```
> Customers2 <-read.csv("~/UniversalBank.csv",header = T)
> ##need to exclude columns ID and Zip
> Customers2<-Customers2[,c(-1,-5)]
>
> # as.factor
> Customers2$Personal.Loan<-as.factor(Customers2$Personal.Loan)
> Customers2$Education<-as.factor(Customers2$Education)
> Customers2$Family<-as.factor(Customers2$Family)
> Customers2$Securities.Account<-as.factor(Customers2$Securities.Account)
> Customers2$CD.Account<-as.factor(Customers2$CD.Account)
> Customers2$CreditCard<-as.factor(Customers2$CreditCard)
> Customers2$Online<-as.factor(Customers2$Online)
>
> trainIndex <- createDataPartition(Customers2$Personal.Loan, p=.500,list = FALSE, ti
mes = 1)
> Customers.test <- Customers2[trainIndex,]
> Customers1 <- Customers2[-trainIndex,]
>
> trainIndex1<-createDataPartition(Customers1$Personal.Loan, p=.600,list = FALSE, tim
es = 1)
> Customers.train <- Customers1[trainIndex1,]
> Customers.valid <- Customers1[-trainIndex1,]
>
> norm.test <- preProcess(Customers.test, method=c("center", "scale"))
>
```

```
>
> Customers.trainnorm.target<-Customers.train$Personal.Loan
> Customers.trainnorm<-predict(norm.test,Customers.train[, -8])
>
> test.knn <- cbind(Customers.testnorm.z, Customers.test$Personal.Loan)
> test.knn.predictors <- test.knn[, 1:11]
> test.knn.target <- test.knn[,12]
>
>
> #confusionMatrix
> preds.k <- class::knn (train=Customers.trainnorm, test=test.knn.predictors, cl=Cust
omers.trainnorm.target, k=1, prob=TRUE)
> confusionMatrix(preds.k, test.knn.target, positive="1")
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 2066  226
         1  194   14

               Accuracy : 0.832
                 95% CI : (0.8168, 0.8465)
    No Information Rate : 0.904
    P-Value [Acc > NIR] : 1.0000

                  Kappa : -0.0293

 Mcnemar's Test P-Value : 0.1304

            Sensitivity : 0.05833
            Specificity : 0.91416
         Pos Pred Value : 0.06731
         Neg Pred Value : 0.90140
             Prevalence : 0.09600
         Detection Rate : 0.00560
   Detection Prevalence : 0.08320
      Balanced Accuracy : 0.48625

       'Positive' Class : 1
```

میبینم دقت این مدل از مدل قبلی ما که داده‌ها را **60** به ۴۰ تقسیم کردیم کمتر است.دقت این مدل برابر۰.۸۳۲ است.

که بنظر میرسد دلیل این کاهش به تقسیم داده‌ها برمیگردد، ما دراینجا برای آموزش مدل از ۵۰درصد داده‌ها استفاده کردیم ولی در مدل قبلی از ۶۰درصد داده‌ها استفاده کردیم