Blood Donations

Bridget Bertoni

April 12, 2018

```
# read in data
train_data=read.csv(file="/Users/bbertoni/Desktop/github/Blood_Donations/training_data.csv",header=T)
test_data=read.csv(file="/Users/bbertoni/Desktop/github/Blood_Donations/test_data.csv",header=T)

names(train_data)=c("ID","Last_donation","Num_donations","Volume","First_donation","Made_donation")
names(test_data)=c("ID","Last_donation","Num_donations","Volume","First_donation")

head(train_data)
```

```
##    ID Last_donation Num_donations Volume First_donation Made_donation
## 1 619             2            50  12500             98             1
## 2 664             0            13   3250             28             1
## 3 441             1            16   4000             35             1
## 4 160             2            20   5000             45             1
## 5 358             1            24   6000             77             0
## 6 335             4             4   1000              4             0
```

```
head(test_data)
```

```
##    ID Last_donation Num_donations Volume First_donation
## 1 659             2            12   3000             52
## 2 276            21             7   1750             38
## 3 263             4             1    250              4
## 4 303            11            11   2750             38
## 5  83             4            12   3000             34
## 6 500             3            21   5250             42
```

```
# split training data into a training set and a validation set
set.seed(333)
train=sample(1:nrow(train_data),0.7*nrow(train_data),replace=F)
val_data = train_data[-train,]
train_data = train_data[train,]

# check for missing or strange values
sum(is.na(train_data))
```

```
## [1] 0
```

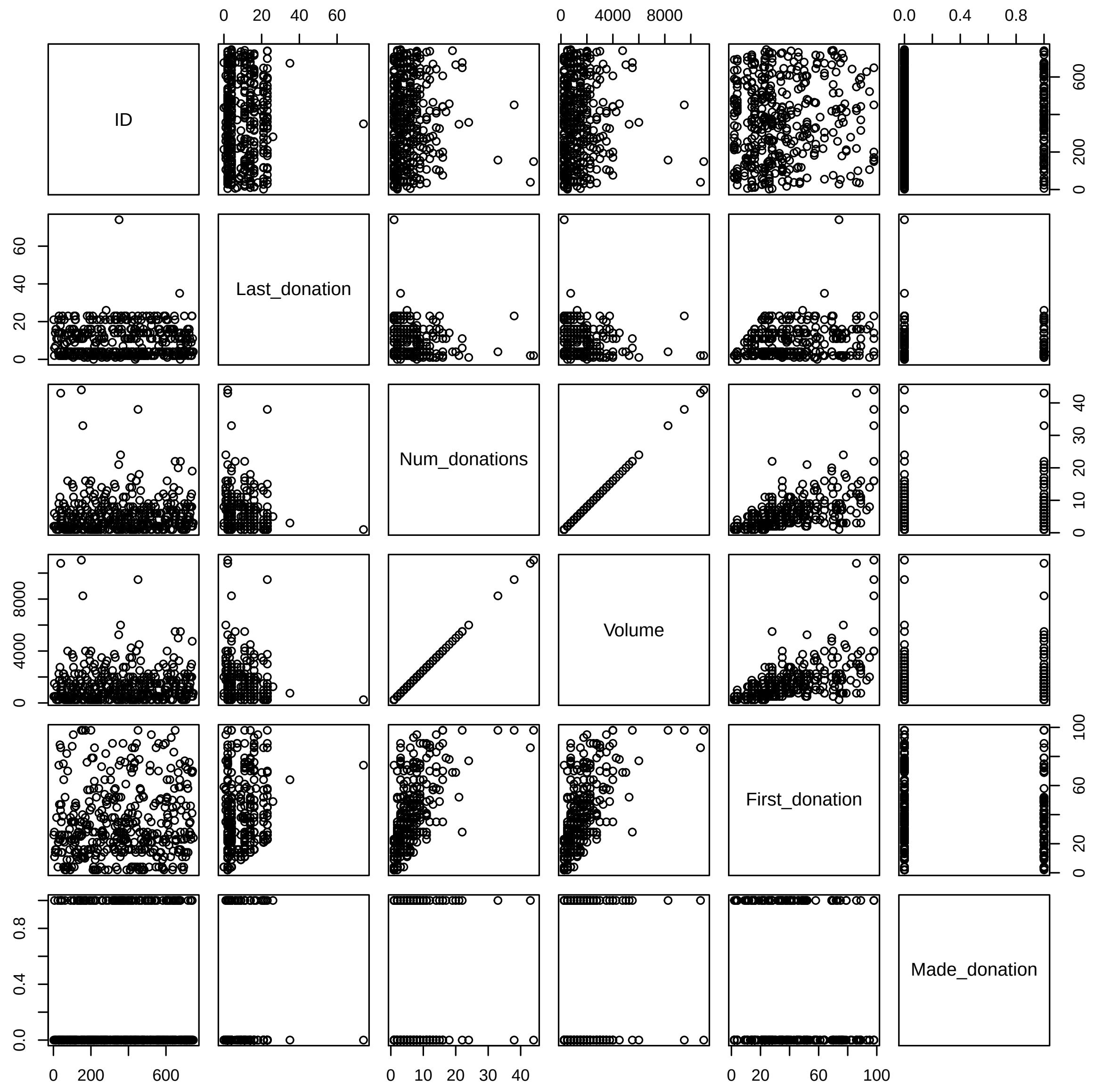
```
sum(is.na(val_data))# no missing values
```

```
## [1] 0
```

```
sum(is.na(test_data)) # missing values
```

```
## [1] 0
```

```
plot(train_data)
```



```r
cor(train_data) # correlation between volume and the number of donations is 1
```

```
##                        ID Last_donation Num_donations      Volume
## ID             1.00000000    0.02180929    0.02167073  0.02167073
## Last_donation  0.02180929    1.00000000   -0.16150193 -0.16150193
## Num_donations  0.02167073   -0.16150193    1.00000000  1.00000000
## Volume         0.02167073   -0.16150193    1.00000000  1.00000000
## First_donation 0.10391260    0.17791940    0.64740429  0.64740429
## Made_donation  0.04101195   -0.24721682    0.20707525  0.20707525
##                First_donation Made_donation
## ID                0.103912600   0.041011949
## Last_donation     0.177919401  -0.247216821
## Num_donations     0.647404293   0.207075245
## Volume            0.647404293   0.207075245
## First_donation    1.000000000   0.008180282
## Made_donation     0.008180282   1.000000000
```

```r
# fit basic logistic regression
glm.fit=glm(Made_donation~Last_donation+Num_donations+First_donation,data=train_data,
            family=binomial)
summary(glm.fit)
```

```
##
## Call:
## glm(formula = Made_donation ~ Last_donation + Num_donations +
##     First_donation, family = binomial, data = train_data)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.3459  -0.7959  -0.5454  -0.3334   2.4440
##
## Coefficients:
##                 Estimate Std. Error z value Pr(>|z|)
## (Intercept)    -0.715501   0.240869  -2.970  0.00297 **
## Last_donation  -0.082632   0.021242  -3.890  0.00010 ***
## Num_donations   0.109057   0.033638   3.242  0.00119 **
## First_donation -0.012572   0.007533  -1.669  0.09513 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 440.16  on 402  degrees of freedom
## Residual deviance: 396.46  on 399  degrees of freedom
## AIC: 404.46
##
## Number of Fisher Scoring iterations: 5
```

```r
#plot(glm.fit)
glm.probs=predict(glm.fit,val_data,type="response")
glm.pred=rep(0,nrow(val_data))
glm.pred[glm.probs>0.5]=1
table(glm.pred,val_data$Made_donation) # confusion matrix
```

```
##
## glm.pred   0   1
##        0 129  38
##        1   1   5
```

```r
mean(glm.pred==val_data$Made_donation)
```

```
## [1] 0.7745665
```

```r
# fit KNN, choose k to minimize the cost on the validation set
library(class)
train.X=as.matrix(train_data,nrow=nrow(train_data),ncol=ncol(train_data))
val.X=as.matrix(val_data,nrow=nrow(val_data),ncol=ncol(val_data))
set.seed(111)
knn.pred=knn(train.X,val.X,train_data$Made_donation,k=1)

table(knn.pred,val_data$Made_donation)
```

```
##
## knn.pred   0   1
##        0 100  30
##        1  30  13
```

```
mean(knn.pred==val_data$Made_donation)
```

```
## [1] 0.6531792
```

```
# fit a random forest
```