ISLR Q8.11 Boosting with Caravan Data

```
library(ISLR)
library(gbm)
```

```
## Loaded gbm 2.1.8
```

11a

Create a training set consisting of the first 1,000 observations, and a test set consisting of the remaining observations.

```
dim(Caravan)
```

```
## [1] 5822   86
```

```
set.seed(1)
train = 1:1000
test = 1001:nrow(Caravan)

Caravan["Purchase"] = ifelse(Caravan$Purchase == "Yes", 1, 0)
# Don't actually use these ???

caravan.train = Caravan[train,]
caravan.test = Caravan[-train,]
caravan.train.y = Caravan[train,"Purchase"]
caravan.test.y = Caravan[-train,"Purchase"]
```

11b

Fit a boosting model to the training set with Purchase as the response and the other variables as predictors.

- Use 1,000 trees, and a shrinkage value of 0.01.
- Which predictors appear to be the most important?

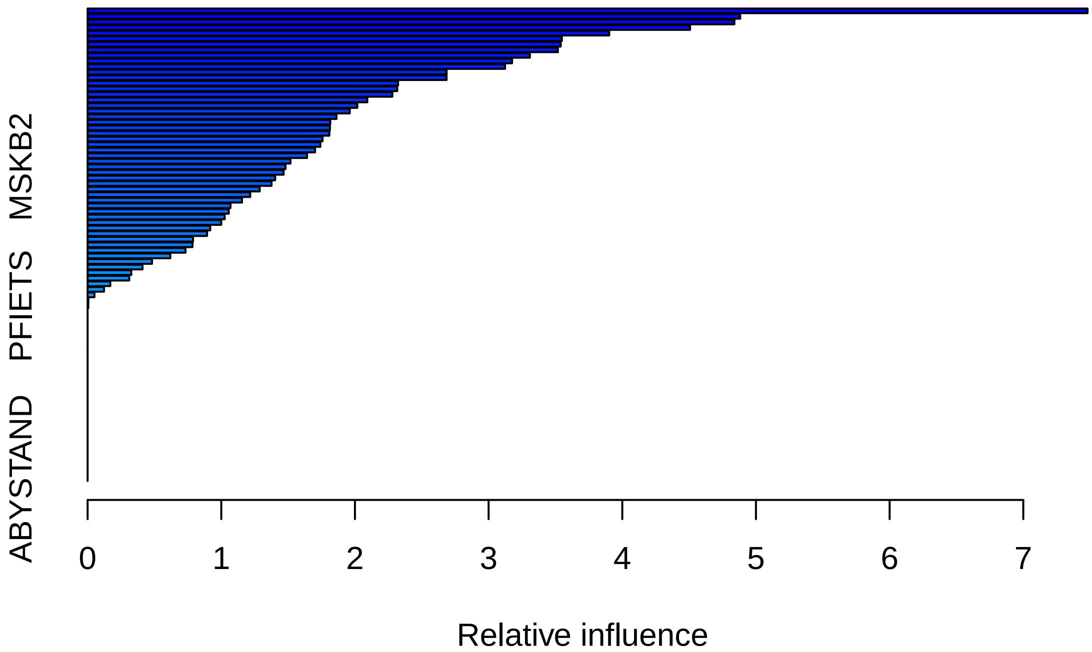
Bernoulli for classification. Gaussian for regression.

```
boost.caravan=gbm(Purchase ~ ., data=Caravan[train,],
                  distribution="bernoulli",
                  n.trees=1000,
                  shrinkage=0.01,
                  interaction.depth=4,
                  verbose=F)
```

```
## Warning in gbm.fit(x = x, y = y, offset = offset, distribution = distribution, :
## variable 50: PVRAAUT has no variation.
```

```
## Warning in gbm.fit(x = x, y = y, offset = offset, distribution = distribution, :
## variable 71: AVRAAUT has no variation.
```

```
summary(boost.caravan)
```




```
##                var     rel.inf
## PPERSAUT PPERSAUT 7.480819014
## MOPLHOOG MOPLHOOG 4.882054338
## MGODGE     MGODGE 4.838869962
## MKOOPKLA MKOOPKLA 4.507280400
## MOSTYPE   MOSTYPE 3.902338079
## MGODPR     MGODPR 3.547892360
## PBRAND     PBRAND 3.539487907
## MBERMIDD MBERMIDD 3.518082698
## MBERARBG MBERARBG 3.309004843
## MINK3045 MINK3045 3.175313873
## MSKC         MSKC 3.123008472
## MSKA         MSKA 2.685844523
## MAUT2       MAUT2 2.685548007
## MAUT1       MAUT1 2.322786246
## PWAPART   PWAPART 2.316252267
## MSKB1       MSKB1 2.279820190
## MRELOV     MRELOV 2.092410309
## MFWEKIND MFWEKIND 2.017651081
## MBERHOOG MBERHOOG 1.961378700
## MBERARBO MBERARBO 1.862074416
## MRELGE     MRELGE 1.815276446
## MINK7512 MINK7512 1.812894054
## MINKM30   MINKM30 1.808781053
## MOPLMIDD MOPLMIDD 1.757784665
## MFGEKIND MFGEKIND 1.741172971
## MGODOV     MGODOV 1.701539077
## MZFONDS   MZFONDS 1.641658796
## MFALLEEN MFALLEEN 1.517763739
## MSKB2       MSKB2 1.480397941
## MINK4575 MINK4575 1.466410983
## MAUT0       MAUT0 1.403097259
## ABRAND     ABRAND 1.375696683
```

```
## MHHUUR     MHHUUR 1.287672857
## MINKGEM   MINKGEM 1.216351643
## MHKOOP     MHKOOP 1.154970948
## MGEMLEEF MGEMLEEF 1.068800262
## MGODRK     MGODRK 1.056066524
## MRELSA     MRELSA 1.025383382
## MZPART     MZPART 0.999705745
## MSKD         MSKD 0.917077921
## MGEMOMV   MGEMOMV 0.893757812
## MBERZELF MBERZELF 0.788935429
## APERSAUT APERSAUT 0.784652995
## MOPLLAAG MOPLLAAG 0.732210597
## MOSHOOFD MOSHOOFD 0.618703929
## PMOTSCO   PMOTSCO 0.481824116
## PLEVEN     PLEVEN 0.410808274
## PBYSTAND PBYSTAND 0.326851643
## MBERBOER MBERBOER 0.311571820
## MINK123M MINK123M 0.169710044
## MAANTHUI MAANTHUI 0.122660387
## ALEVEN     ALEVEN 0.051158218
## PAANHANG PAANHANG 0.006040057
## PFIETS     PFIETS 0.004694048
## PWABEDR   PWABEDR 0.000000000
## PWALAND   PWALAND 0.000000000
## PBESAUT   PBESAUT 0.000000000
## PVRAAUT   PVRAAUT 0.000000000
## PTRACTOR PTRACTOR 0.000000000
## PWERKT     PWERKT 0.000000000
## PBROM       PBROM 0.000000000
## PPERSONG PPERSONG 0.000000000
## PGEZONG   PGEZONG 0.000000000
## PWAOREG   PWAOREG 0.000000000
## PZEILPL   PZEILPL 0.000000000
## PPLEZIER PPLEZIER 0.000000000
## PINBOED   PINBOED 0.000000000
## AWAPART   AWAPART 0.000000000
## AWABEDR   AWABEDR 0.000000000
## AWALAND   AWALAND 0.000000000
## ABESAUT   ABESAUT 0.000000000
## AMOTSCO   AMOTSCO 0.000000000
## AVRAAUT   AVRAAUT 0.000000000
## AAANHANG AAANHANG 0.000000000
## ATRACTOR ATRACTOR 0.000000000
## AWERKT     AWERKT 0.000000000
## ABROM       ABROM 0.000000000
## APERSONG APERSONG 0.000000000
## AGEZONG   AGEZONG 0.000000000
## AWAOREG   AWAOREG 0.000000000
## AZEILPL   AZEILPL 0.000000000
## APLEZIER APLEZIER 0.000000000
## AFIETS     AFIETS 0.000000000
## AINBOED   AINBOED 0.000000000
## ABYSTAND ABYSTAND 0.000000000
```

Predict the Training Data

```
train.predict.prob = predict.gbm(boost.caravan, newdata = Caravan[train,], n.trees = 1000)
train.predict = ifelse(train.predict.prob > 0.5, 1, 0)
```

Confusion Matrix

```
table(caravan.train.y, train.predict)
```

```
##                train.predict
## caravan.train.y   0   1
##               0 941   0
##               1  49  10
```

Calculate Training Classification Accuracy

```
(941+10)/1000
```

```
## [1] 0.951
```

11c Predict the Test Data

Use the boosting model to predict the response on the test data.

- Predict that a person will make a purchase if the estimated probability of purchase is greater than **20%**.
- Form a confusion matrix.
- What fraction of the people predicted to make a purchase do in fact make one? **90%**
- How does this compare with the results obtained from applying KNN or logistic regression to this data set? **In Chapter 4 Lab, KNN and LR produced much worse results. $<$ 35% accuracy**

```
test.predict.prob = predict.gbm(boost.caravan,
                                newdata = Caravan[-train,],
                                n.trees = 1000,
                                type = "response")
test.predict = ifelse(test.predict.prob > 0.2, 1, 0)
```

Confusion Matrix

```
table(caravan.test.y, test.predict)
```

```
##               test.predict
## caravan.test.y    0    1
##              0 4336  197
##              1  258   31
```

Calculate Test Classification Accuracy

```
(4336 + 31)/4822
```

```
## [1] 0.9056408
```