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?

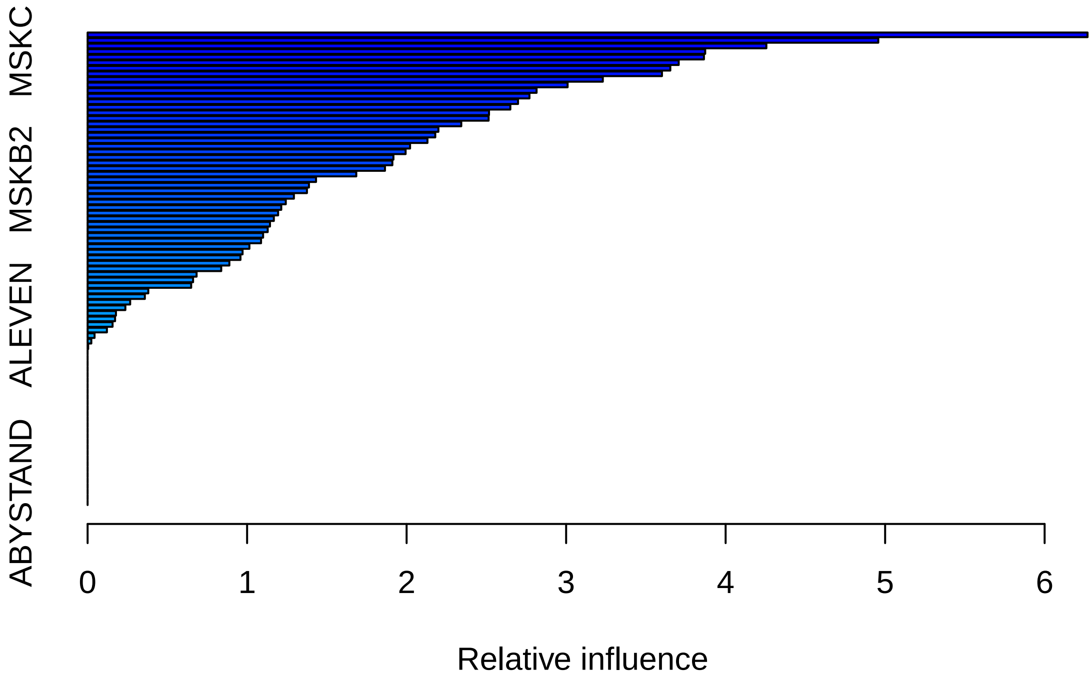
Bernoulli for classification. Gaussian for regression.

```
boost.caravan=gbm(Purchase ~ ., data=Caravan[train,],
                  distribution="bernoulli",
                  n.trees=1000,
                  shrinkage=0.1,
                  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 6.269132745
## MOSTYPE   MOSTYPE 4.957208947
## MINK3045 MINK3045 4.255481718
## MSKC         MSKC 3.871943679
## MGODPR     MGODPR 3.864119953
## MAUT2       MAUT2 3.706009310
## PBRAND     PBRAND 3.653207279
## MKOOPKLA MKOOPKLA 3.600915531
## MGODGE     MGODGE 3.230076925
## ABRAND     ABRAND 3.009404492
## MBERARBG MBERARBG 2.814735114
## MOPLHOOG MOPLHOOG 2.770135208
## MBERHOOG MBERHOOG 2.698629622
## MSKB1       MSKB1 2.650323411
## MSKA         MSKA 2.516577936
## MAUT1       MAUT1 2.514163821
## MRELOV     MRELOV 2.342722409
## MBERMIDD MBERMIDD 2.199435273
## MRELSA     MRELSA 2.179739793
## MINKM30   MINKM30 2.130652061
## MBERARBO MBERARBO 2.021640077
## MAUT0       MAUT0 1.993664399
## PWAPART   PWAPART 1.917324407
## MRELGE     MRELGE 1.910577071
## MZFONDS   MZFONDS 1.864316002
## MSKD         MSKD 1.684470682
## MSKB2       MSKB2 1.432148348
## MHKOOP     MHKOOP 1.387700696
## MFALLEEN MFALLEEN 1.375055905
## MGEMLEEF MGEMLEEF 1.293317355
## MFWEKIND MFWEKIND 1.242630639
```

```
## MOPLMIDD MOPLMIDD 1.214053362
## MHHUUR     MHHUUR 1.194454658
## MBERZELF MBERZELF 1.167720560
## MZPART     MZPART 1.143357717
## MGODOV     MGODOV 1.129158825
## APERSAUT APERSAUT 1.100349770
## MGODRK     MGODRK 1.087390244
## MINK7512 MINK7512 1.014352996
## MFGEKIND MFGEKIND 0.971495973
## MINK4575 MINK4575 0.958335680
## MGEMOMV   MGEMOMV 0.888709703
## PFIETS     PFIETS 0.837805368
## MINKGEM   MINKGEM 0.683261064
## MINK123M MINK123M 0.661311502
## MOPLLAAG MOPLLAAG 0.649365148
## PLEVEN     PLEVEN 0.380600799
## PMOTSCO   PMOTSCO 0.358987881
## MOSHOOFD MOSHOOFD 0.266905017
## MBERBOER MBERBOER 0.236770387
## MAANTHUI MAANTHUI 0.177777408
## PBROM       PBROM 0.172198638
## ALEVEN     ALEVEN 0.156522983
## PBYSTAND PBYSTAND 0.121121370
## PTRACTOR PTRACTOR 0.043668668
## PAANHANG PAANHANG 0.023309538
## PWALAND   PWALAND 0.003553936
## PWABEDR   PWABEDR 0.000000000
## PBESAUT   PBESAUT 0.000000000
## PVRAAUT   PVRAAUT 0.000000000
## PWERKT     PWERKT 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   4  55
```

Calculate Training Classification Accuracy

```
(941+55)/1000
```

```
## [1] 0.996
```

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 prob- ability 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?
- How does this compare with the results obtained from applying KNN or logistic regression to this data set?

```
test.predict.prob = predict.gbm(boost.caravan, newdata = Caravan[-train,], n.trees = 1000, type = "respo
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 4339  194
##              1  255   34
```

Calculate Test Classification Accuracy

```
(4339 + 34)/4822
```

```
## [1] 0.9068851
```