H4_S21_P2

Naveen

4/27/2021

Problem 2

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

```
caravan.ds<-read.csv("HW4_S21/CARAVAN.csv")
caravan.ds$Purchase<-ifelse(caravan.ds$Purchase=="Yes",1,0)
#bernoulli doens't work unless the Response variable is of class character
caravan.ds$Purchase<-as.character(caravan.ds$Purchase)
caravan.ds<-caravan.ds%>% mutate(RID = row_number())
caravan.ds.train <- caravan.ds %>% dplyr::slice_head(n=1000)
caravan.ds.test  <- anti_join(caravan.ds, caravan.ds.train, by = 'RID')
caravan.ds.train<-caravan.ds.train %>% dplyr::select(-c(RID))
caravan.ds.test<-caravan.ds.test %>% dplyr::select(-c(RID))
#knn doesn't work when scaled values are nan, creating seperate frame for knn
caravan.ds.train.scaled<- data.frame(caravan.ds.train %>% select(-c(Purchase)) %>% scale)
caravan.ds.test.scaled<- data.frame(caravan.ds.test %>% select(-c(Purchase)) %>% scale)
caravan.ds.train.scaled$PVRAAUT<-ifelse(is.nan(caravan.ds.train.scaled$PVRAAUT),0,0)
caravan.ds.train.scaled$AVRAAUT<-ifelse(is.nan(caravan.ds.train.scaled$AVRAAUT),0,0)
```

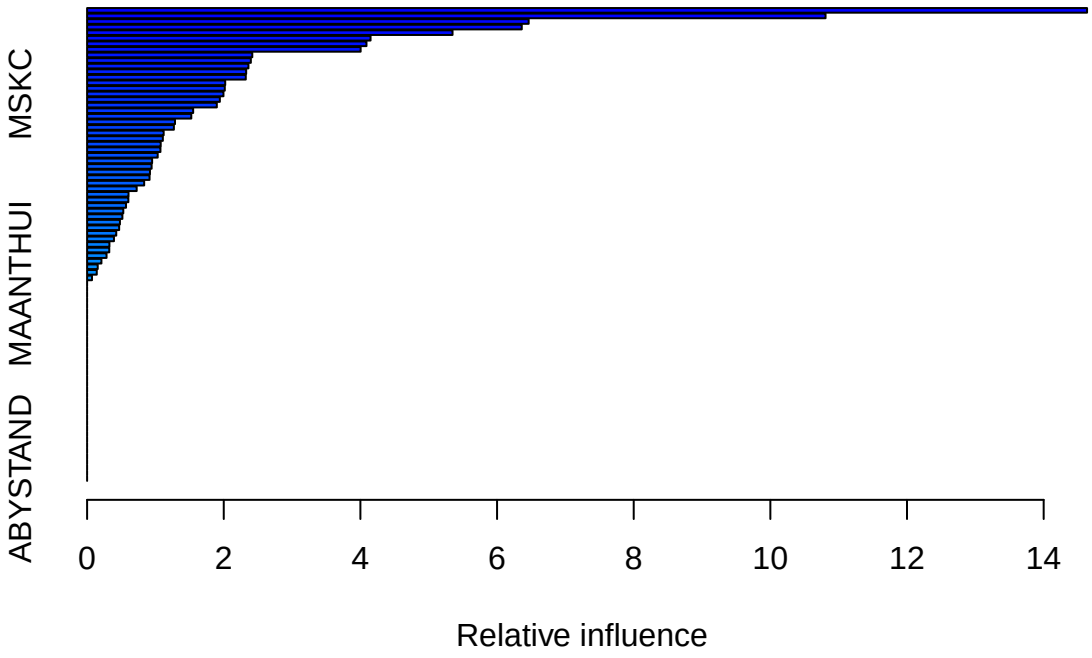
b) Fit a boosting model to the training set with Purchase as the response and the other variables as predictors. Use 1,000 trees, and a shrinkage value of 0.01. Which predictors appear to be the most important?

```
set.seed(5)
boost.caravan.ds.fit<-gbm(Purchase~.,data = caravan.ds.train, distribution = "bernoulli", n.trees = 100
                      shrinkage = 0.01)
```

```
## 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.
```

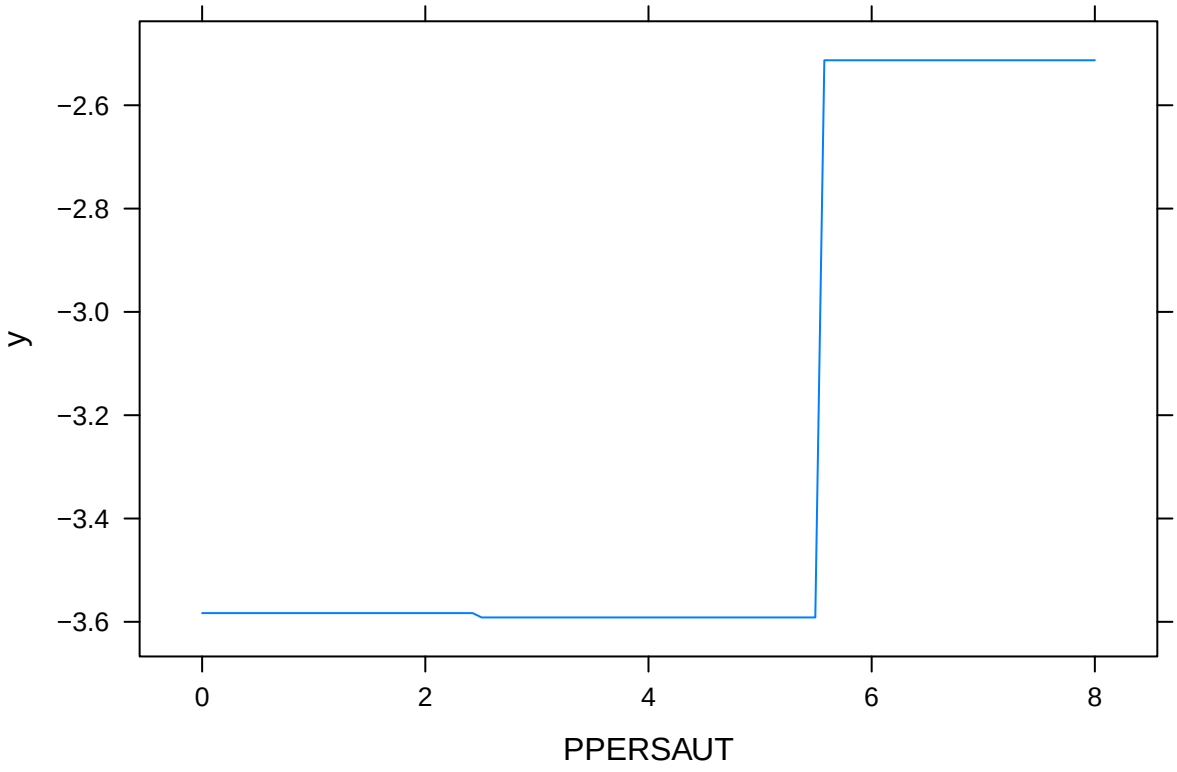
```
boost.caravan.ds.predict<-predict(boost.caravan.ds.fit, newdata = caravan.ds.test,n.trees = 1000, type =
predicted.purchase<-ifelse(boost.caravan.ds.predict>0.2,"1","0")
summary(boost.caravan.ds.fit)
```



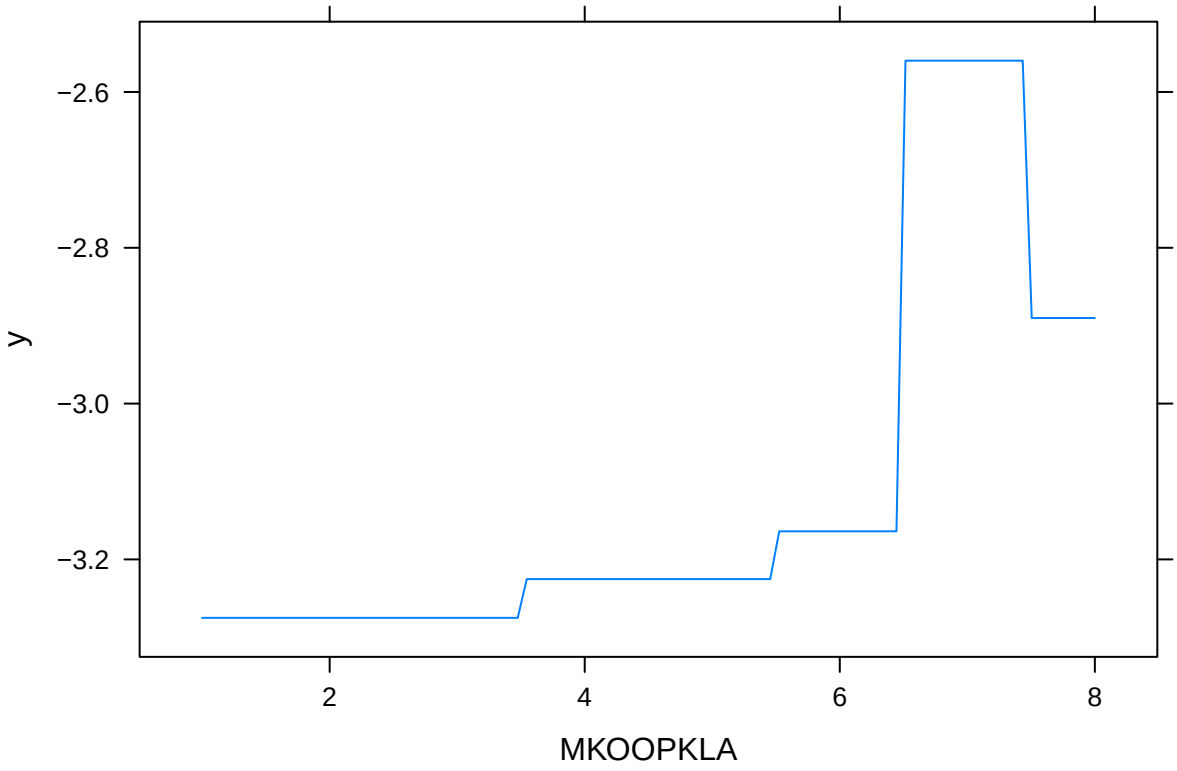

```
##                var     rel.inf
## PPERSAUT PPERSAUT 14.63519385
## MKOOPKLA MKOOPKLA 10.80775869
## MOPLHOOG MOPLHOOG  6.46281343
## MBERMIDD MBERMIDD  6.36141845
## PBRAND     PBRAND  5.34828459
## MGODGE     MGODGE  4.14859078
## ABRAND     ABRAND  4.08888390
## MINK3045 MINK3045  4.00327299
## PWAPART   PWAPART  2.41736909
## MSKA         MSKA  2.39635505
## MINKGEM   MINKGEM  2.36151432
## MAUT2       MAUT2  2.32796089
## MGODPR     MGODPR  2.32223079
## MAUT1       MAUT1  2.02121827
## MOSTYPE   MOSTYPE  2.01530148
## MSKC         MSKC  1.99578439
## MBERHOOG MBERHOOG  1.94304406
## MBERARBG MBERARBG  1.89850680
## PBYSTAND PBYSTAND  1.55239075
## MRELGE     MRELGE  1.52497218
## MINK7512 MINK7512  1.28628568
## MGODOV     MGODOV  1.27010632
## MGODRK     MGODRK  1.12061227
## APERSAUT APERSAUT  1.10838638
## MSKD         MSKD  1.07719236
## MSKB1       MSKB1  1.07315282
## MOPLMIDD MOPLMIDD  1.03311174
## MAUT0       MAUT0  0.95142058
## MINKM30   MINKM30  0.94409509
## MFWEKIND MFWEKIND  0.91979519
## MFGEKIND MFGEKIND  0.91420410
```

```
## MINK4575 MINK4575  0.83510909
## MRELOV     MRELOV  0.72566461
## MOSHOOFD MOSHOOFD  0.60620604
## MHHUUR     MHHUUR  0.60380352
## MHKOOP     MHKOOP  0.56934690
## MBERBOER MBERBOER  0.52970179
## MZPART     MZPART  0.51652596
## MBERARBO MBERARBO  0.48041153
## PMOTSCO   PMOTSCO  0.46916473
## PLEVEN     PLEVEN  0.42654929
## MGEMLEEF MGEMLEEF  0.39318771
## MGEMOMV   MGEMOMV  0.32657396
## MRELSA     MRELSA  0.32447332
## MZFONDS   MZFONDS  0.28439837
## MOPLLAAG MOPLLAAG  0.20951055
## MSKB2       MSKB2  0.15533586
## MINK123M MINK123M  0.14129531
## MFALLEEN MFALLEEN  0.07151417
## MAANTHUI MAANTHUI  0.00000000
## MBERZELF MBERZELF  0.00000000
## PWABEDR   PWABEDR  0.00000000
## PWALAND   PWALAND  0.00000000
## PBESAUT   PBESAUT  0.00000000
## PVRAAUT   PVRAAUT  0.00000000
## PAANHANG PAANHANG  0.00000000
## PTRACTOR PTRACTOR  0.00000000
## PWERKT     PWERKT  0.00000000
## PBROM       PBROM  0.00000000
## PPERSONG PPERSONG  0.00000000
## PGEZONG   PGEZONG  0.00000000
## PWAOREG   PWAOREG  0.00000000
## PZEILPL   PZEILPL  0.00000000
## PPLEZIER PPLEZIER  0.00000000
## PFIETS     PFIETS  0.00000000
## PINBOED   PINBOED  0.00000000
## AWAPART   AWAPART  0.00000000
## AWABEDR   AWABEDR  0.00000000
## AWALAND   AWALAND  0.00000000
## ABESAUT   ABESAUT  0.00000000
## AMOTSCO   AMOTSCO  0.00000000
## AVRAAUT   AVRAAUT  0.00000000
## AAANHANG AAANHANG  0.00000000
## ATRACTOR ATRACTOR  0.00000000
## AWERKT     AWERKT  0.00000000
## ABROM       ABROM  0.00000000
## ALEVEN     ALEVEN  0.00000000
## APERSONG APERSONG  0.00000000
## AGEZONG   AGEZONG  0.00000000
## AWAOREG   AWAOREG  0.00000000
## AZEILPL   AZEILPL  0.00000000
## APLEZIER APLEZIER  0.00000000
## AFIETS     AFIETS  0.00000000
## AINBOED   AINBOED  0.00000000
## ABYSTAND ABYSTAND  0.00000000
```

```
#marginal dependence plot
plot(boost.caravan.ds.fit,i="PPERSAUT")
```




```
plot(boost.caravan.ds.fit,i="MKOOPKLA")
```



car policies(PPERSAUT) and Purchasing power class(MKOOPKLA) are far the most important variables.

c) 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? How does this compare with the results obtained from applying KNN or logistic regression to this data set?

Boosting

```
caravan.ds.test <- data.frame(caravan.ds.test,predicted.purchase)
caravan.ds.test$PurchaseYN<-ifelse(caravan.ds.test$Purchase=="1","Yes","No")
caravan.ds.test$predicted.purchase<-ifelse(caravan.ds.test$predicted.purchase=="1","Yes","No")
caravan.ds.cm<- table(data.frame(caravan.ds.test$PurchaseYN, caravan.ds.test$predicted.purchase))
caravan.ds.cm
```

```
##                           caravan.ds.test.predicted.purchase
## caravan.ds.test.PurchaseYN   No  Yes
##                        No  4410  123
##                        Yes  254   35
```

```
1 - ( (caravan.ds.cm[1,1] + caravan.ds.cm[2,2]) / nrow(caravan.ds.test))
```

```
## [1] 0.07818333
```

Knn

```
knn.predicted.purchase<-knn(caravan.ds.train.scaled,  caravan.ds.test.scaled,caravan.ds.train$Purchase,
caravan.ds.test <- data.frame(caravan.ds.test,knn.predicted.purchase)
caravan.ds.test$knn.predicted.purchase<-ifelse(caravan.ds.test$knn.predicted.purchase=="1","Yes","No")
caravan.ds.knn.cm<- table(data.frame(caravan.ds.test$PurchaseYN, caravan.ds.test$knn.predicted.purchase
caravan.ds.knn.cm
```

```
##                           caravan.ds.test.knn.predicted.purchase
## caravan.ds.test.PurchaseYN   No  Yes
##                        No  4254  279
##                        Yes  249   40
```

```
1 - ( (caravan.ds.knn.cm[1,1] + caravan.ds.knn.cm[2,2]) / nrow(caravan.ds.test))
```

```
## [1] 0.1094981
```

Error rate with boosting is 0.07818333 and knn is 0.1094981