Analysis_of_demographics_information

Bowen Xiao

May 22, 2018

Data Preparation

Demographics dataset has 6627 records and about 84% are negative cases. I split the data into train set (80%) and test set (20%).

```
setwd("~/Parkinson_Classification_Based_on_Demographics_Information_and_Voice_Features")
parkinson <- read.csv("./data/parkinson.csv")
parkinson$brain<-as.factor(parkinson$brain)
parkinson$edu<-as.factor(parkinson$edu)
parkinson$emp<-as.factor(parkinson$emp)
parkinson$gender<-as.factor(parkinson$gender)
parkinson$mar<-as.factor(parkinson$mar)
parkinson$race<-as.factor(parkinson$race)
parkinson$smoke<-as.factor(parkinson$smoke)
parkinson$diag<-as.factor(parkinson$diag)
nrow(parkinson[parkinson$diag=='FALSE',])/nrow(parkinson)
```

```
## [1] 0.8367285
```

```
set.seed(123)
index=sample(1:nrow(parkinson),0.8*nrow(parkinson))
parkinson_train<-parkinson[index,]
parkinson_test<-parkinson[-index,]
```

Logistic Regression

```
model1<-glm(diag~.,data=parkinson_train,family=binomial(link='logit'))
y_1<-predict.glm(model1,newdata = parkinson_test,type='response')
y_1=ifelse(y_1>=0.5,'TRUE','FALSE')
A1<-mean(y_1==parkinson_test$diag)
R1<-mean(y_1[parkinson_test$diag=='TRUE']==parkinson_test[parkinson_test$diag=='TRUE',]$diag)
```

SVM

```
library(e1071)
svmfit<-svm(diag~.,data=parkinson_train,kernel="radial")
svmpred<-predict(svmfit,newdata = parkinson_test)
A2<-mean(svmpred==parkinson_test$diag)
R2<-mean(svmpred[parkinson_test$diag=='TRUE']==parkinson_test[parkinson_test$diag=='TRUE',]$diag)
```

Naive Bayes Network

```
library(mlbench)
naive <- naiveBayes(diag ~ ., data = parkinson_train)
y_naive<-predict(naive,newdata = parkinson_test)
A3<-mean(y_naive==parkinson_test$diag)
R3<-mean(y_naive[parkinson_test$diag=='TRUE']==parkinson_test[parkinson_test$diag=='TRUE',]$diag)
```

For example, the marginal distribution of `gender` is shown as following.

```
library(knitr)
kable(naive$tables$gender)
```

	Female	Male	Prefer not to answer	UNK
FALSE	0.1846572	0.8128536	0.0013578	0.0011315
TRUE	0.3492063	0.6496599	0.0000000	0.0011338

Random Forest

```
library(randomForest)
```

```
## randomForest 4.6-14
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
fit_rf<-randomForest(diag~.,data = parkinson_train)
rfpred<-predict(fit_rf,newdata = parkinson_test)
A4<-mean(rfpred==parkinson_test$diag)
R4<-mean(rfpred[parkinson_test$diag=='TRUE']==parkinson_test[parkinson_test$diag=='TRUE',]$diag)
```

XGBoost

```
library(xgboost)
data_train<-model.matrix(~.+0,data = parkinson_train[,1:8])
data_test<-model.matrix(~.+0,data = parkinson_test[,1:8])
dtrain <- xgb.DMatrix(data = data_train,label = ifelse(parkinson_train$diag=='TRUE',1,0))
dtest <- xgb.DMatrix(data = data_test,label = ifelse(parkinson_test$diag=='TRUE',1,0))
params <- list(booster = "gbtree", objective = "binary:logistic", eta=0.3, gamma=0,
               max_depth=6, min_child_weight=1, subsample=1, colsample_bytree=1)
xgbcv <- xgb.cv( params = params, data = dtrain, nrounds = 100, nfold = 5, showsd = T,
                 stratified = T, print.every.n = 10, early.stop.round = 20, maximize = F)
```

```
## Warning: 'print.every.n' is deprecated.
## Use 'print_every_n' instead.
## See help("Deprecated") and help("xgboost-deprecated").
```

```
## Warning: 'early.stop.round' is deprecated.
## Use 'early_stopping_rounds' instead.
## See help("Deprecated") and help("xgboost-deprecated").
```

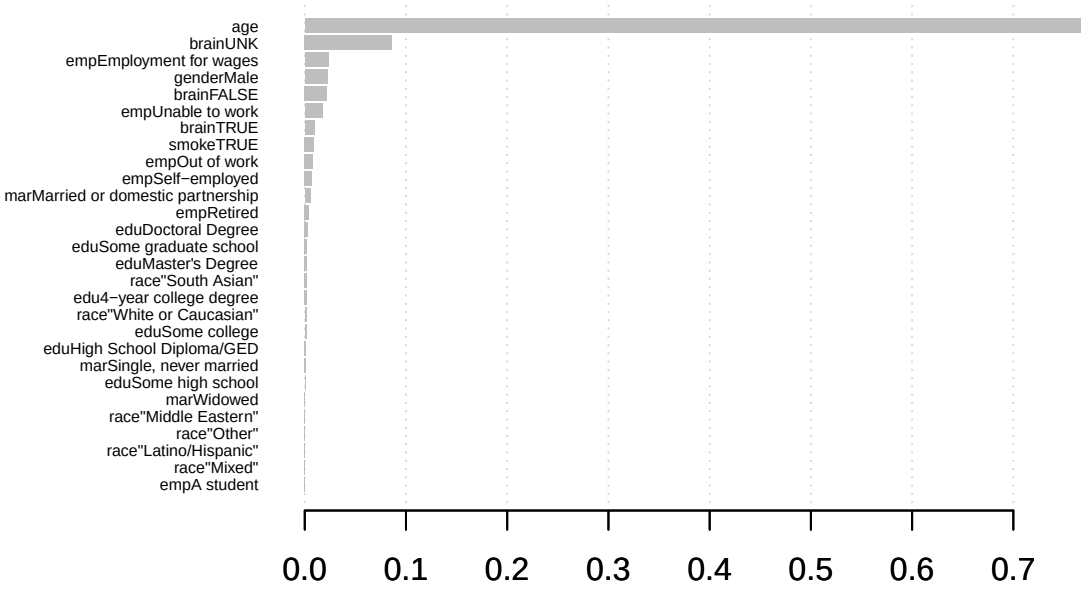
```
## [1]  train-error:0.080409+0.001496   test-error:0.088663+0.004622
## Multiple eval metrics are present. Will use test_error for early stopping.
## Will train until test_error hasn't improved in 20 rounds.
##
```

```
## [11] train-error:0.073335+0.000882   test-error:0.083380+0.004447
## [21] train-error:0.070223+0.001446   test-error:0.083947+0.002877
## Stopping. Best iteration:
## [8]  train-error:0.073760+0.000912   test-error:0.082060+0.004547
```

```
fit_xgb<-xgb.train(data = dtrain, max_depth = 6, eta = 0.3, nthread = 2, nrounds = 11,
                   objective = "binary:logistic")
xgpred<-predict(fit_xgb,newdata = dtest)
A5<-mean(ifelse(xgpred<=0.5,0,1)==ifelse(parkinson_test$diag=='TRUE',1,0))
R5<-mean(ifelse(xgpred<=0.5,0,1)[parkinson_test$diag=='TRUE']==
           ifelse(parkinson_test$diag=='TRUE',1,0)[parkinson_test$diag=='TRUE'])
```

Importance of each predictor is shown as following.

```
mat <- xgb.importance(feature_names = colnames(data_train),model=fit_xgb)
xgb.plot.importance (importance_matrix = mat)
```



Summary

Comparison of the 5 methods is shown as following.

```
kable(data.frame(method=c('logistic','SVM','Naive Bayes','Random Forest','XGBoost'),
                 Accuracy=c(A1,A2,A3,A4,A5),
                 Recall=c(R1,R2,R3,R4,R5)))
```

method	Accuracy	Recall
logistic	0.9147813	0.685
SVM	0.9125189	0.635
Naive Bayes	0.9087481	0.810
Random Forest	0.9162896	0.740
XGBoost	0.9147813	0.750

Combining with Voice Features - Logistic Regression with Regularization/Lasso

Combined dataset is decoded with dummy variables.

```
setwd("~/Parkinson_Classification_Based_on_Demographics_Information_and_Voice_Features")
train <- read.csv("./src/R/train.csv", header=FALSE)
test <- read.csv("./src/R/test.csv", header=FALSE)
mean(train[,1]==1)
```

```
## [1] 0.6270381
```

Penalty parameter λ is chosed based on cross validation.

```
library(glmnet)
```

```
## Loading required package: Matrix
```

```
## Loading required package: foreach
```

```
## Loaded glmnet 2.0-16
```

```
lasso_cv<-cv.glmnet(x=as.matrix(train[,-1]),y=train[,1],alpha = 1,family="binomial")
model2<-glmnet(x=as.matrix(train[,-1]),y=train[,1],alpha = 1,family="binomial",
                    lambda = lasso_cv$lambda.min)
(A<-mean(predict(model2,newx=as.matrix(test[,-1]),type="class")==test[,1]))
```

```
## [1] 0.8929512
```

```
(R<-mean(predict(model2,newx=as.matrix(test[,-1]),type="class")[test[,1]==0]==test[test[,1]==0,1]))
```

```
## [1] 0.7878657
```