Depression Patient Outcome Study

Lingge Li

11/21/2016

Patient outcome

Most patients are moderately or severely depressed before treatment (BDI>30) and mildly depressed after (BDI<20).

```
load('Depression.RData')
Depression <- Depression[Depression$GROUP_ID==2,]
summary(Depression$Pre_BDI)
```

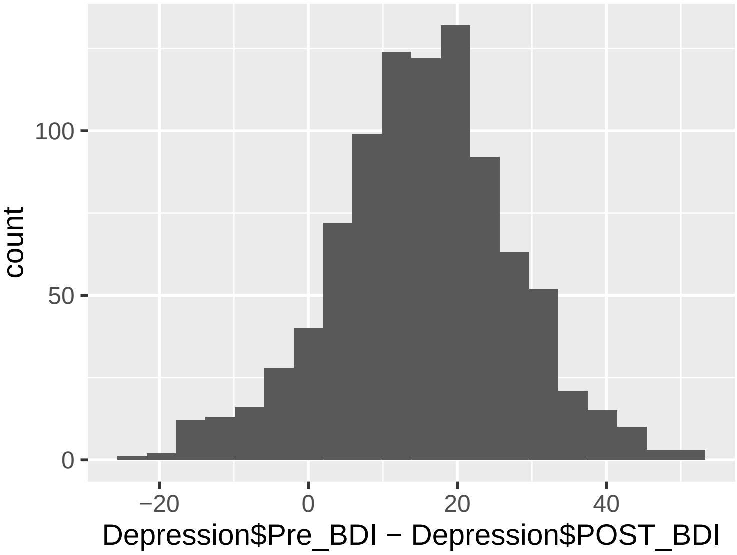
```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##      20      24      30      31      36      60
```

```
summary(Depression$POST_BDI)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     0.0     6.0    14.0    15.9    22.0    60.0
```

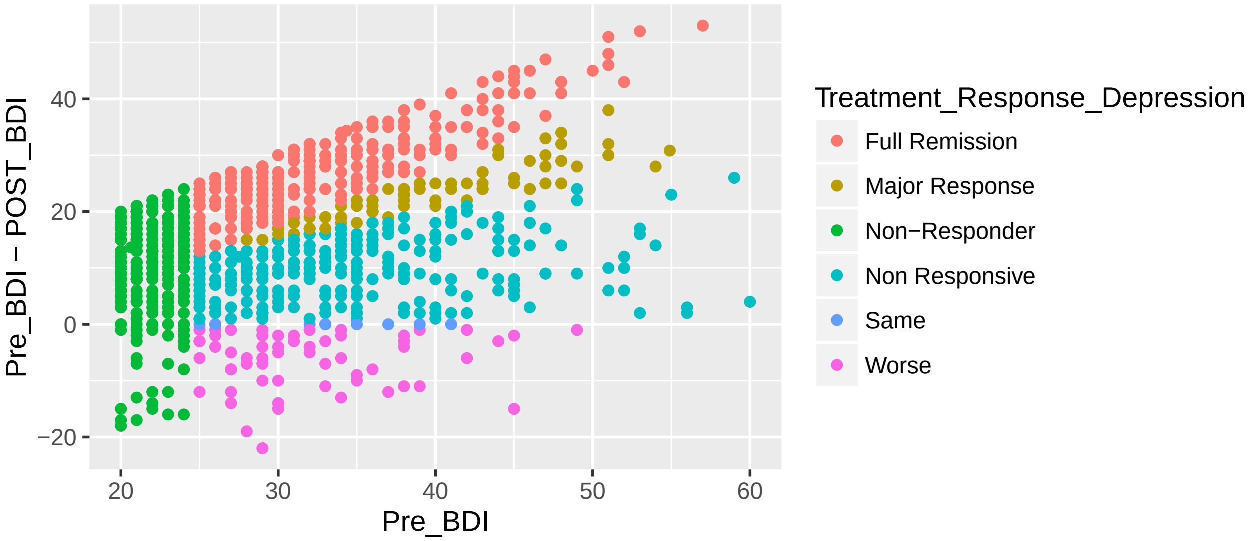
Changes in BDI are mostly positive.

```
qplot(Depression$Pre_BDI-Depression$POST_BDI, bins=20)
```



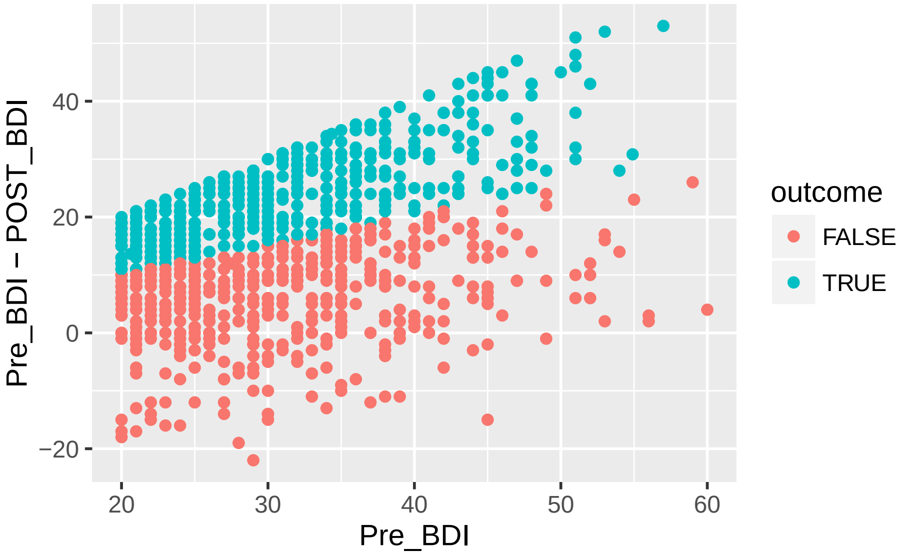
In the original data, all patients who have low BDI to start with are considered non-responders, which is inconsistent with our discussion.

```
ggplot(data=Depression,aes(x=Pre_BDI,y=Pre_BDI-POST_BDI,colour=Treatment_Response_Depression)) + geom_p
```



We consider patients who improve BDI by more than 50% as responders.

```
outcome <- (Depression$Pre_BDI-Depression$POST_BDI)>0.5*Depression$Pre_BDI
temp <- data.frame(Pre_BDI=Depression$Pre_BDI,POST_BDI=Depression$POST_BDI,outcome)
ggplot(data=temp,aes(x=Pre_BDI,y=Pre_BDI-POST_BDI,colour=outcome)) + geom_point()
```

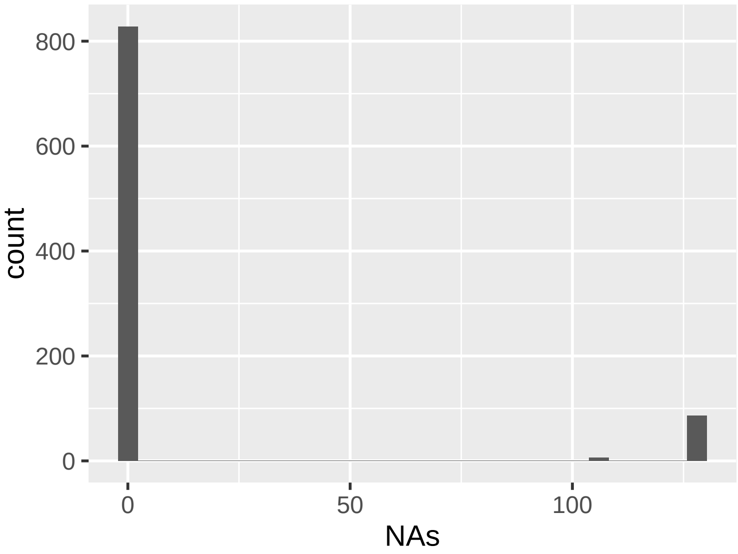


Brain regions

Here we first look at brain region scores of baseline SPECT scan. Most patients have 1 or 2 missing values but about 90 patients have more than 100 missing values. We will remove those patients and do simple mean imputation for the rest.

```
NAs <- apply(Depression[,125:252], 1, function(x) sum(is.na(x)))
qplot(NAs)
```

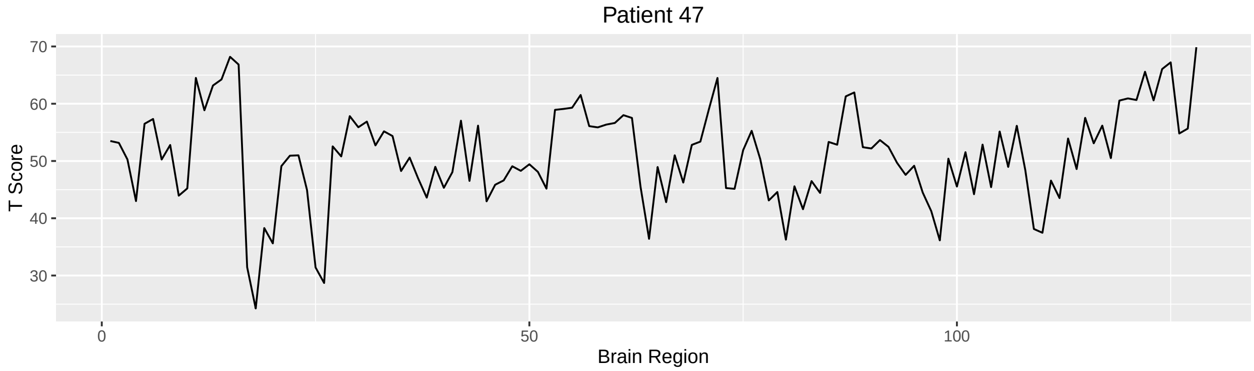
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



```
Depression <- Depression[NAs<100,]
Depression[,'T_Baseline_Cerebellum_10_L'][is.na(Depression[,'T_Baseline_Cerebellum_10_L'])] <- mean(Depr
Depression[,'T_Baseline_Cerebellum_10_R'][is.na(Depression[,'T_Baseline_Cerebellum_10_R'])] <- mean(Depr
Depression[,'T_Baseline_Cerebellum_3_L'][is.na(Depression[,'T_Baseline_Cerebellum_3_L'])] <- mean(Depres
Depression[,'T_Baseline_Vermis_1_2'][is.na(Depression[,'T_Baseline_Vermis_1_2'])] <- mean(Depression[,'T
Depression[,'T_Baseline_Vermis_10'][is.na(Depression[,'T_Baseline_Vermis_10'])] <- mean(Depression[,'T_B
```

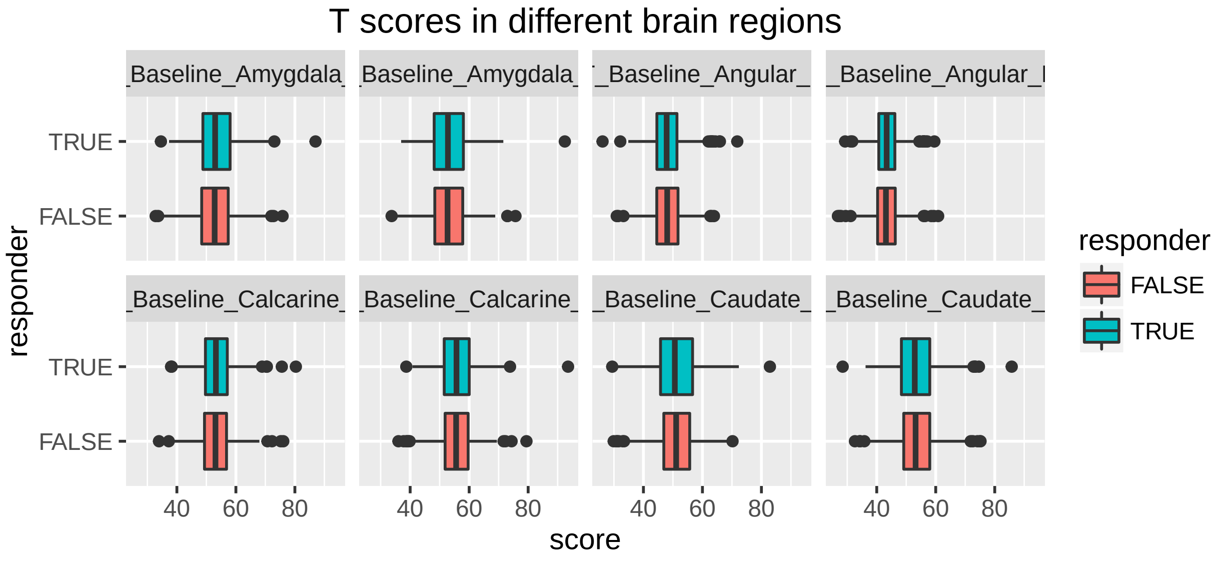
Let's plot the brain region scores of one patient as a curve. There is quite a bit of fluctuation across regions.

```
baseline <- Depression[,125:252]
b <- as.numeric(baseline[47,])
x <- seq(1,128,1)
qplot(x,b,geom='line',xlab='Brain Region',ylab='T Score',main=paste('Patient 47'))
```



Side by side boxplots show almost no difference between responders and non-responders, which is bad.

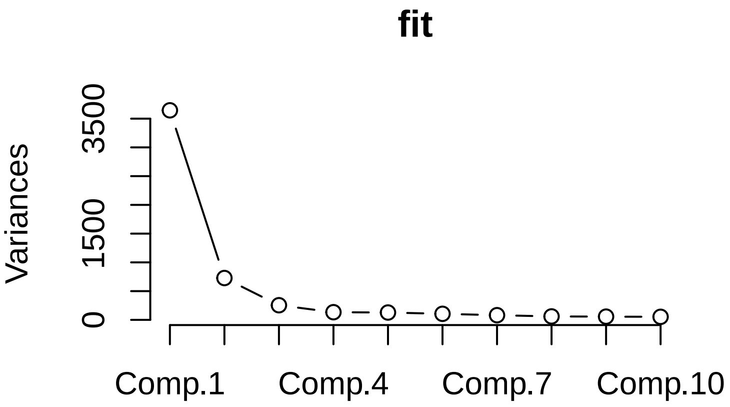
```
stackregion <- function(regions, outcome) {
  n <- length(outcome)
  score <- rep(NA, n*8)
  name <- rep(NA, n*8)
  responder <- rep(NA, n*8)
  patient <- rep(NA, n*8)
  for (i in 1:8) {
    score[((i-1)*n+1):((i-1)*n+n)] <- as.numeric(regions[,i])
    name[((i-1)*n+1):((i-1)*n+n)] <- colnames(regions)[i]
    responder[((i-1)*n+1):((i-1)*n+n)] <- outcome
    patient[((i-1)*n+1):((i-1)*n+n)] <- seq(1, n, 1)
  }
  stacked <- data.frame(score,name,responder,patient)
  return(stacked)
}
outcome <- (Depression$Pre_BDI-Depression$POST_BDI)>0.5*Depression$Pre_BDI
i <- 1
regions <- stackregion(Depression[,(125+(i-1)*8):(132+(i-1)*8)],outcome)
ggplot(data=regions,aes(x=responder,y=score,fill=responder)) +
  geom_boxplot() + coord_flip() +
  facet_wrap(~name, ncol=4) +
  theme(aspect.ratio=3/4) +
  ggtitle(label='T scores in different brain regions')
```



Dimensionality reduction

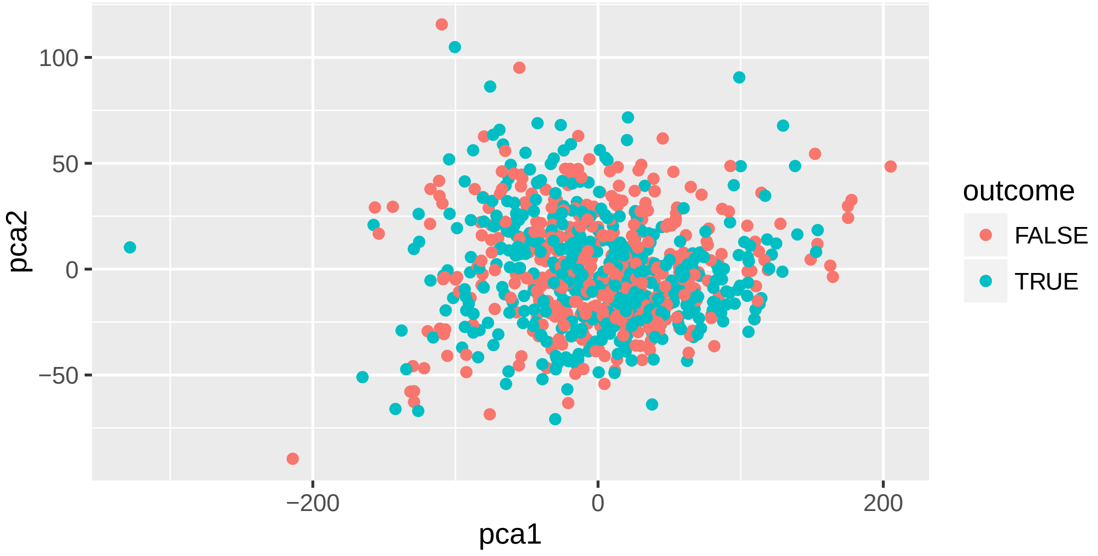
Since there are 128 brain regions, it would be helpful to identify the important ones. The first two principal components account for most of the variance.

```
fit <- princomp(baseline, cor=FALSE)
plot(fit,type="lines")
```



However, the data do not look separable, which is expected.

```r
pca1 <- fit$scores[,1]
pca2 <- fit$scores[,2]
pca <- data.frame(pca1, pca2, outcome)
ggplot(data=pca, aes(x=pca1, y=pca2, colour=outcome)) + geom_point()
```



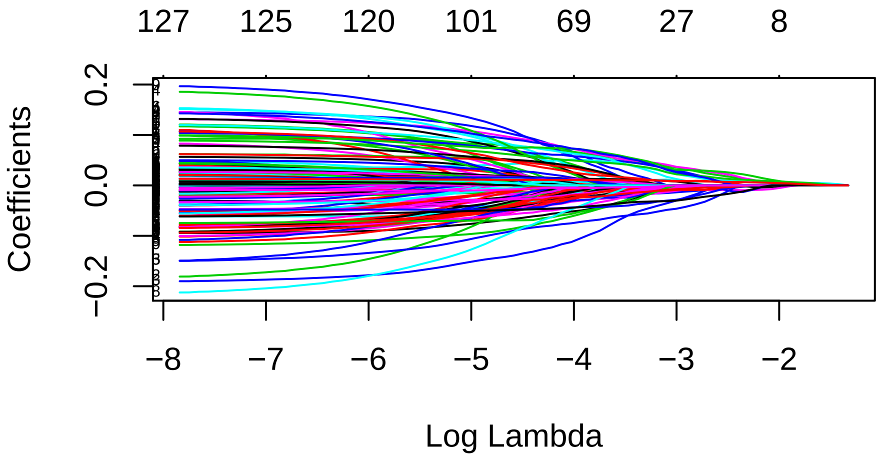
Classification models

We will go ahead to build classification models regardless. First, penalized logistic regression. Shrinkage does not improve prediction accuracy.

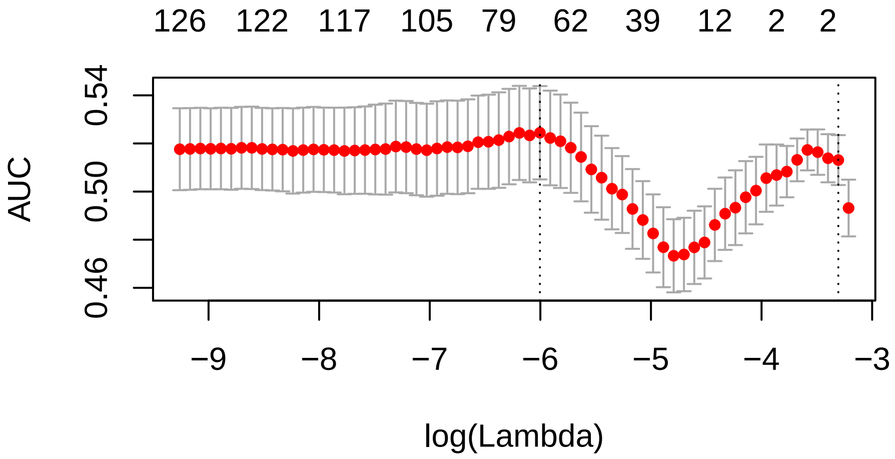
```r
library(glmnet)
```

```
## Warning: package 'glmnet' was built under R version 3.2.4

## Loading required package: Matrix

## Loading required package: foreach

## Loaded glmnet 2.0-5
```

```
X <- as.matrix(cbind(baseline, Pre_BDI=Depression$Pre_BDI))

indices <- sample(1:828,828)
Xtest <- X[indices[1:160],]
ytest <- outcome[indices[1:160]]
Xtrain <- X[indices[161:828],]
ytrain <- outcome[indices[161:828]]

fit <- glmnet(Xtrain, as.factor(ytrain), family='binomial', standardize=FALSE)
plot(fit, xvar ='lambda', label=TRUE)
```



```
cvfit = cv.glmnet(X, outcome, family='binomial', type.measure='auc')
plot(cvfit)
```



Then support vector machine and gradient boosted decision trees with 5-fold cross-validation. None of them does well.

```
library(e1071)
library(xgboost)
```

```
## Warning: package 'xgboost' was built under R version 3.2.5
```

```r
dat <- cbind(outcome,baseline,Depression$Pre_BDI)
indices <- sample(1:828,800)
dat <- dat[indices,]
accuracy1 <- 0
accuracy2 <- 0
for (i in 1:5) {
  current <- seq(1+(i-1)*160,160+(i-1)*160,1)
  train <- dat[-current,]
  test <- dat[current,]
  fit1 <- svm(as.factor(outcome)~.,data=train)
  pred1 <- predict(fit1,test[,-1],type='class')
  accuracy1 <- accuracy1+sum(test$outcome==pred1)
  fit2 <- xgboost(data=as.matrix(train[,-1]),label=train$outcome,nrounds=10,max.depth=5,
                 objective='binary:logistic',verbose=0)
  pred2 <- predict(fit2, as.matrix(test[,-1]))
  accuracy2 <- accuracy2+sum(test$outcome==(pred2>0.5))
}
accuracy1/800
```

```
## [1] 0.49375
```

```r
accuracy2/800
```

```
## [1] 0.48375
```

Of course all this is just the beginning and fairly standard. One could do more detailed exploratory analysis to generate new features, use other models with fine-tuned hyperparameters and ensemble the results. Also, note that I did most of data manipulation with base R functions because the data size is small.