Classification

Classification Example

Here we use a low dimensional dataset that tries to build a predictive model for presence of infarct-like lesions (as measured by MRI) on individuals over the age of 65. We select a subset of all available features on these patients and attempt to build a predictive model for infarcts.

We first select a subset of features to use in our predictive model:

```r
set.seed(10)

dat.infarc <- read.table("infarc-data/infarcts.txt", header = T)

dat.infarc.use <- dat.infarc %>%
  select(infarcts, age, educ, income, weight, height, packyrs, alcoh,
         chd, claud, htn, diabetes, ldl, crt) %>%
  na.omit()

train.ind <- sample(1:nrow(dat.infarc.use), floor(nrow(dat.infarc.use)/2))

dat.train <- dat.infarc.use[train.ind,]
dat.test <- dat.infarc.use[-train.ind,]

fit <- glm(infarcts~ .,
           family = binomial(),
           data = dat.train)
```

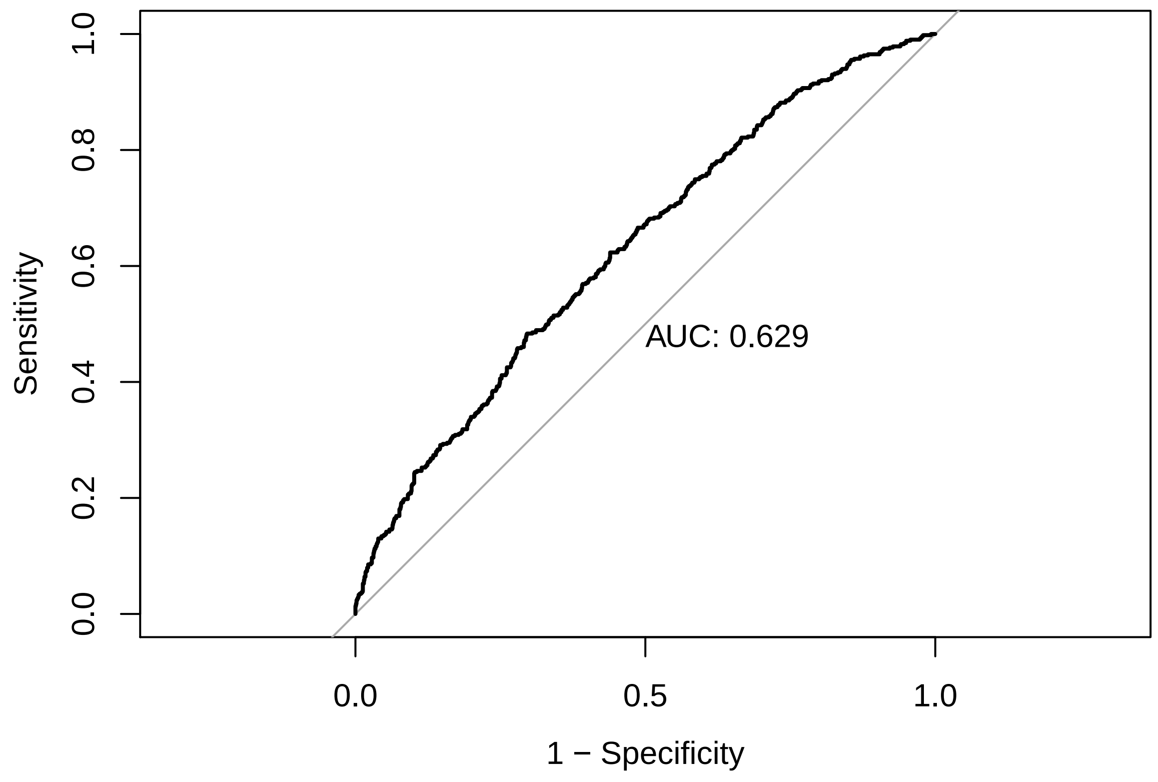
We evaluate goodness of fit using the area under the ROC curve:

```r
#### Evaluating Fit

preds <- predict(fit, dat.test, type = "response")

roc.glm <- roc(dat.test$infarcts, preds)
```

```
## Setting levels: control = 0, case = 1
```

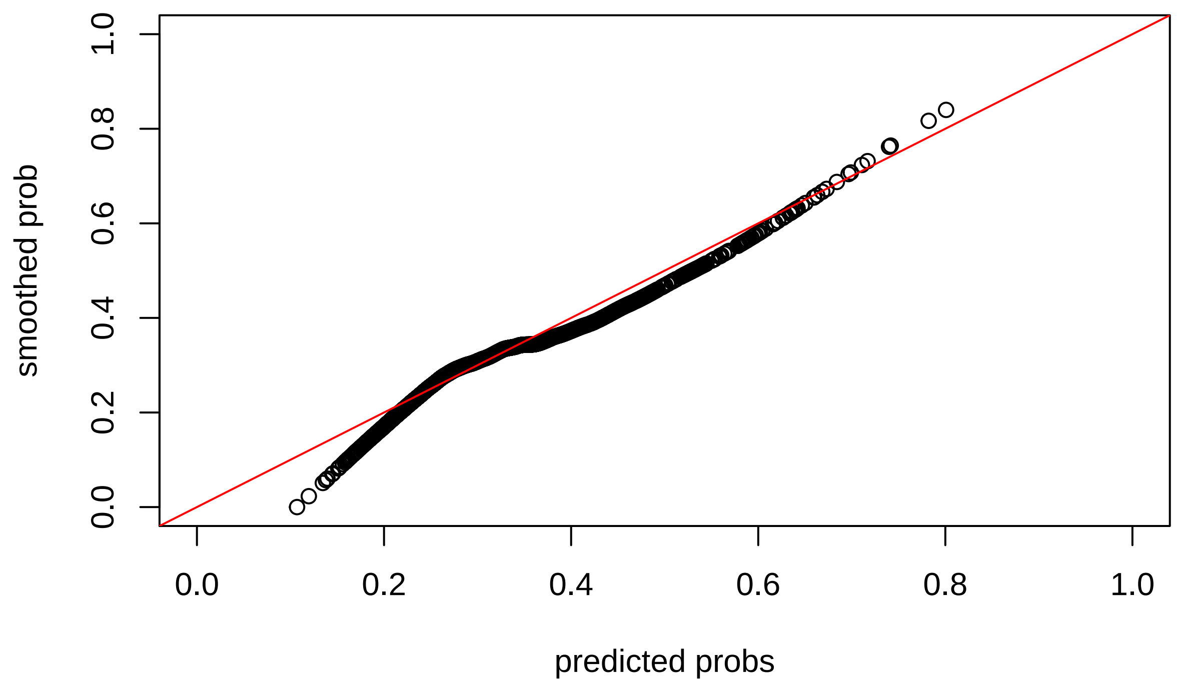
```
## Setting direction: controls < cases
```

```r
plot(roc.glm, legacy.axes = TRUE, print.auc = TRUE)
```



We can also evaluate the calibration of our model

```
### Evaluating calibration

smoother <- loess(dat.test$infarcts ~ preds, span = 0.6, degree = 1)

plot(preds, predict(smoother), xlim = c(0,1), ylim = c(0,1),
     xlab = "predicted probs", ylab = "smoothed prob")
abline(0,1, col = "red")
```



In this case calibration appears to be good.

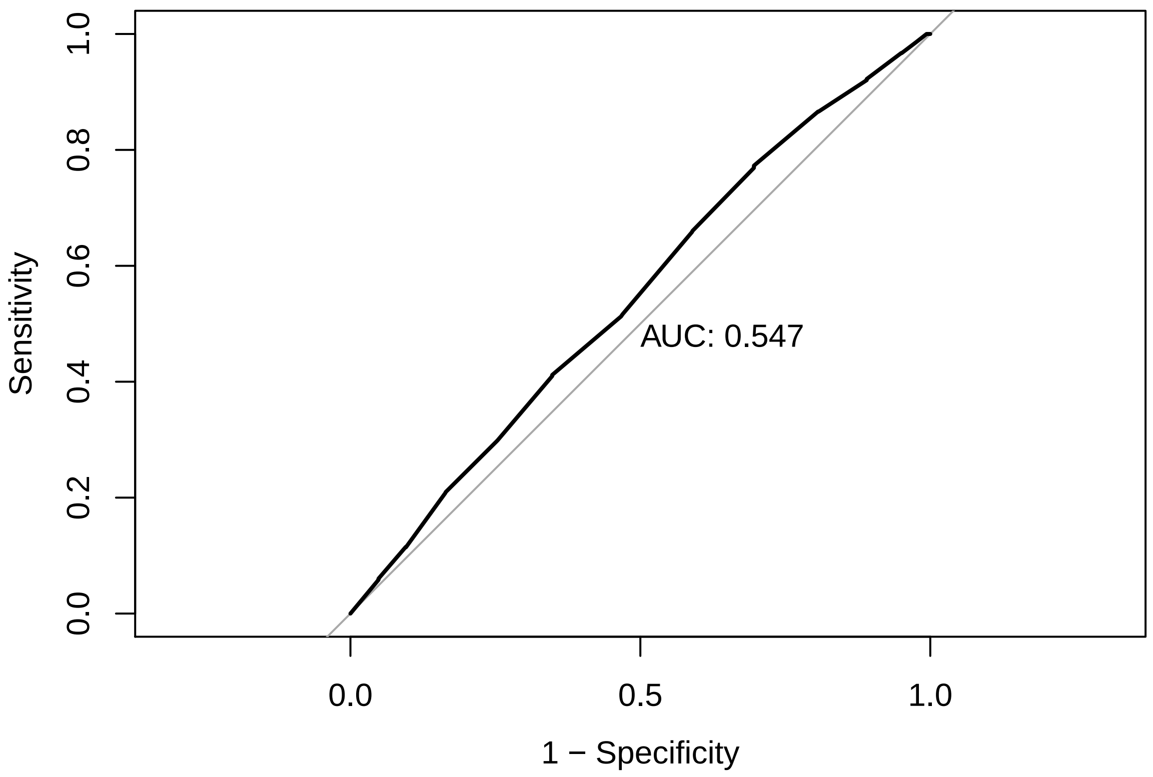
Instead of a simple logistic model we could try something more flexible. In this case we build a nearest-neighbor-based model. Let's try using... say... 35 nearest neighbors.

```
#### Fitting a knn model

preds.35 <- knn(dat.train %>% select(-infarcts),
    dat.test %>% select(-infarcts),
    as.factor(dat.train$infarcts),
    k = 35, prob = TRUE)

#### Evaluating Fit

roc.knn35 <- roc(dat.test$infarcts, attributes(preds.35)$prob)
```

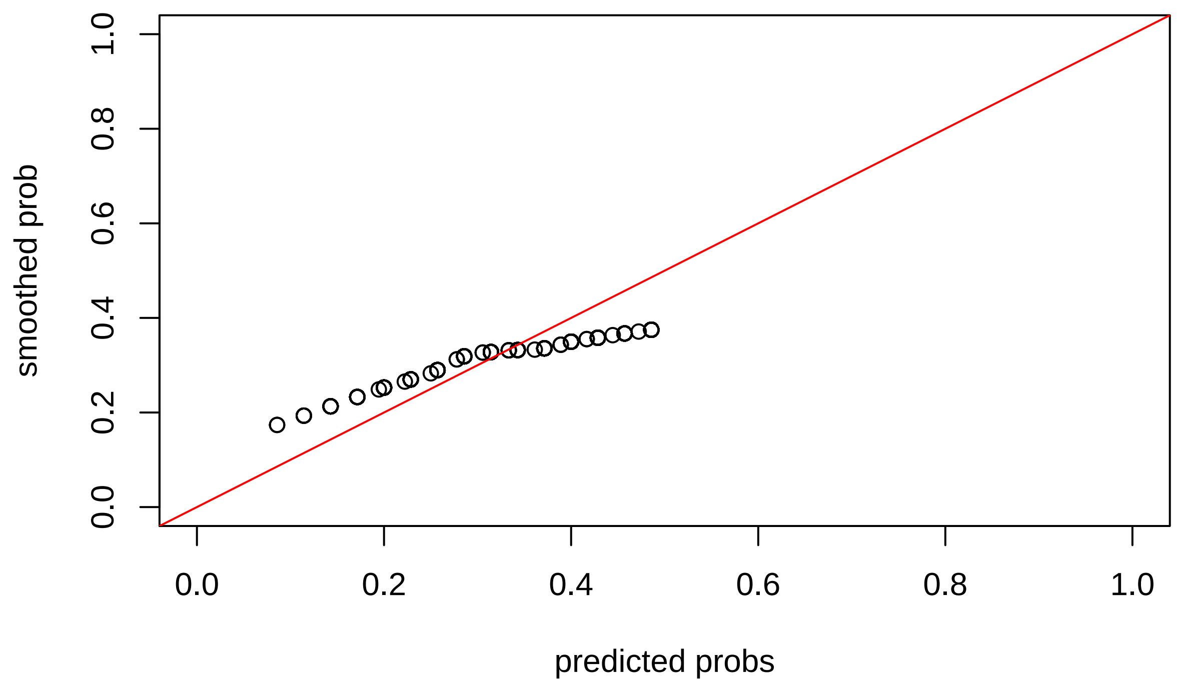
```
## Setting levels: control = 0, case = 1
```

```
## Setting direction: controls > cases
```

```
plot(roc.knn35, legacy.axes = TRUE, print.auc = TRUE)
```



```
### Evaluating calibration

knn35.use <- 1-attributes(preds.35)$prob

smoother <- loess(dat.test$infarcts ~ knn35.use, span = 0.6, degree = 1)

plot(knn35.use, predict(smoother), xlim = c(0,1), ylim = c(0,1), xlab = "predicted probs", ylab = "smoo
abline(0,1, col = "red")
```



Arbitrarily using 35 nearest neighbors was not such a great idea… Let's use split sample validation to select the number of neighbors!

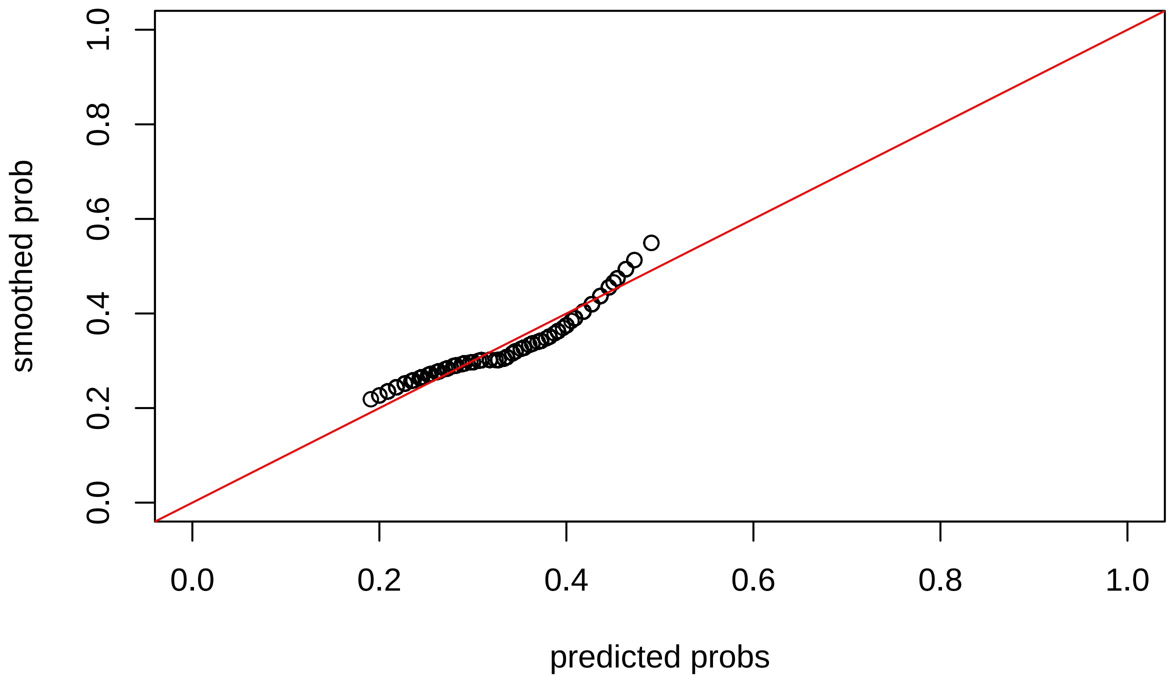
```
### Selecting the number of neighbors

eval.knn <- list()

for(num.k in 1:20){
  preds.knn <- knn(dat.train %>% select(-infarcts),
                   dat.test %>% select(-infarcts),
                   as.factor(dat.train$infarcts),
                   k = 50 + num.k*20, prob = TRUE)
  eval.knn[num.k] <- roc(dat.test$infarcts, attributes(preds.knn)$prob)$auc
}
```

```
unlist(eval.knn)
```

```
##  [1] 0.5590912 0.5593634 0.5570040 0.5605364 0.5656311 0.5663758 0.5668458
##  [8] 0.5691927 0.5646611 0.5652711 0.5643596 0.5684861 0.5696458 0.5683876
## [15] 0.5688948 0.5694020 0.5705927 0.5638090 0.5584838 0.5580973
```

```
num.k.use <- which.min(unlist(eval.knn)) * 20 + 50

preds.knn <- knn(dat.train %>% select(-infarcts),
                 dat.test %>% select(-infarcts),
                 as.factor(dat.train$infarcts),
                 k = num.k.use, prob = TRUE)

## Evaluating calibration

preds.knn.use <- 1-attributes(preds.knn)$prob ## flipping things because knn flipped class label

smoother <- loess(dat.test$infarcts ~ preds.knn.use, span = 0.6, degree = 1)
```

```r
plot(preds.knn.use, predict(smoother), xlim = c(0,1), ylim = c(0,1),
     xlab = "predicted probs", ylab = "smoothed prob")
abline(0,1, col = "red")
```



That looks a little better!