ECON7333: Assignment 2

Thomas Doyle

Setup

```
options(scipen=999)
set.seed(1234)
LR2 <- read.table(file="./LR2.csv", header = TRUE, sep = ",")
attach(LR2)
```

Exercise 1

Consider a probit model given by the following conditional probability,

$$\Pr(Y = 1|X = x) = \Phi(\beta_0 + \beta_1 x)$$

Where Φ is the cumulative distribution function of a standard normal random variable, $\mathcal{N}(0, 1)$:

$$\Phi(x) = \int_{-\infty}^{x} \frac{1}{\sqrt{2\pi}} \exp^{-\frac{1}{2}t^2}\,dt$$

Thus $\Phi(\beta_0 + \beta_1 x) = \mathrm{P}(Z \leq z),\ Z \sim \mathcal{N}(0, 1)$.

Write an R function that computes the maximum likelihood estimate along with bootstrapped errors.

```
probit_mle_b <- function(x,y,opt=NULL) {

  opts <- opt # Optional predict vector

  # probit link
  # objective function
  probit <- function(b,x,y) {
    n <- length(y)
    ll <- 0
    for(i in 1:n) {
      z <- b[1]+b[2]*x[i]
      z <- pnorm(z, mean=0, sd=1, log.p = FALSE)
      ll <- ll + log(z)*(y[i]==1) + log(1-z)*(y[i]==0)
    }

    return(-ll)
  }

  # compute the mle
  #
  obj = optim(c(0,0), probit, x=x, y=y)

  coef1 <- obj$par[1]
```

```
  coef2 <- obj$par[2]

  return_list <- list(
    model = obj,
    fitted = pnorm(coef1+coef2*x,0,1),
    coefficients = c(coef1,coef2)
  )

  # Bootstrap the SE
  #

  B <- 100 # boots

  b_boot = matrix(rep(0,2*B),B,2)
  n <- length(y) #n=1000

  # The bootstrap
  for (i in 1:B) {
    # indices for the i-th bootstrap subsample
    ind_ = sample(n,n,replace=TRUE)
    # input vector in the subsample
    xb = x[ind_]
    # output vector in the subsample
    yb = y[ind_]

    # compute the maximum likelihood estimates
    obj = optim(c(0,0), probit, x=xb, y=yb)

    b_boot[i,1] = obj$par[1]
    b_boot[i,2] = obj$par[2]
  }

  return_list$standard_errors <- c(sd(b_boot[,1]),sd(b_boot[,2]))
  return_list$boots <- b_boot


  # Retrun prediction vector if optional new obs are supplied
  #
  if(!is.null(opts)) {
    opts <- unlist(opts)
    if(is.numeric(opts)) {
      return_list$response = ifelse(pnorm(coef1+coef2*opts,0,1)>1/2,1,0)
    }
  }

  return(
    return_list
  )
}
```

We then apply the probit model to the LR2 data set.

```
# Apply probit_mle_bto LR2, supply optional vector x
#
```

```r
est <- probit_mle_b(LR2$x,LR2$y,x)
est$coefficients
```

```
## [1] -4.242606 -3.086030
```

```r
est$standard_errors
```

```
## [1] 0.3382945 0.2723762
```

```r
# Calc test error
mean(est$response!=y)
```

```
## [1] 0.043
```

```r
# Setup train/test hold out, then estimate
train <- LR2[1:800,]
test <- LR2[801:1000,]
est.2 <- probit_mle_b(train$x,train$y,test$x)
est.2$coefficients
```

```
## [1] -4.557228 -3.213275
```

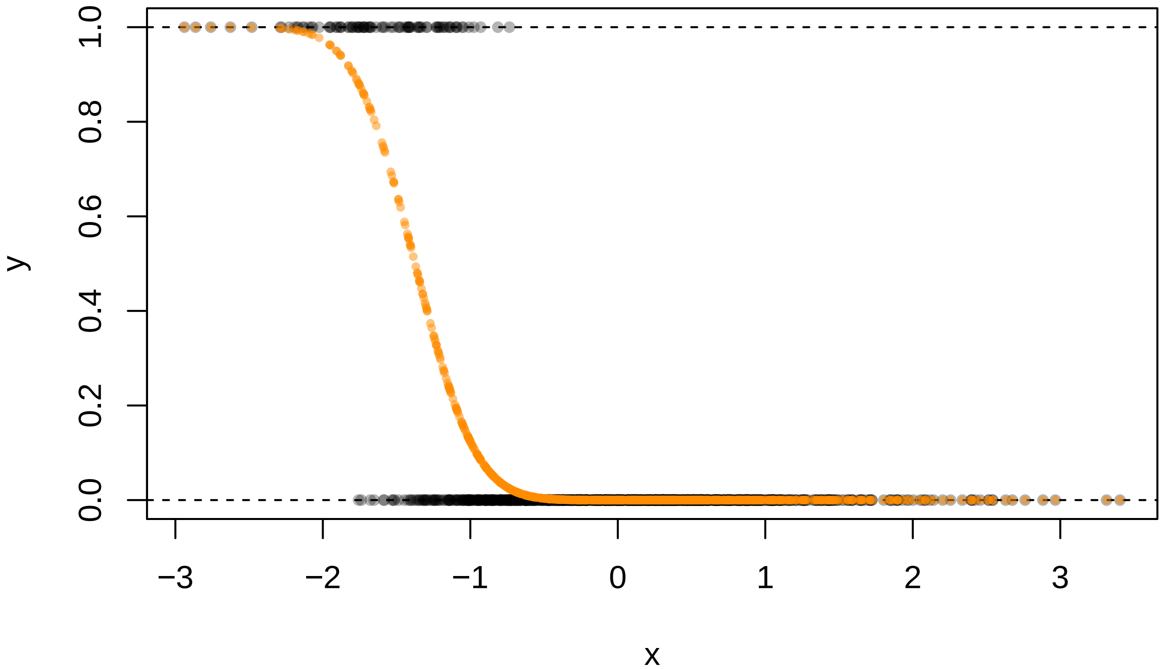
```r
# Calc test error
mean(est.2$response!=test$y)
```

```
## [1] 0.06
```

The `R` function returns a maximum likelihood estimate of the probit model parameters and their estimated standard errors. The estimated parameters are $\beta_0 = -4.232$ (0.304) and $\beta_1 - 3.086$ (0.254) respectively. The `R` function uses a bootstrap sampling method of $n = 1000$ to estimate standard deviation of the fitted parameters. The parameters provide a training error rate of 4.3%, and test error rate of 6.0% when fitted to the `LR2` dataset.

The below figure illustrates a plot of the probit distribution, and predicted response variable, $\hat{y} = \Phi(\beta_0 + \beta_1 x)$.

```r
plot(x,y, pch=20, col=scales::alpha("black",alpha = 0.3), main="")
abline(h=1, lty=2)
abline(h=0, lty=2)
y1 <- sort(est$fitted,TRUE)
points(sort(x),y1,pch=20,cex=0.7,col=scales::alpha("darkorange",0.5))
```



Exercise 2

Consider the model. Let X and U be two independent uniformly distributed random variables and let Y be given by the equation

$$Y = I\left(U \leq \frac{1}{1 + \exp(-\beta_0 - \beta_1 X - \beta_2 X^3 - \beta_3 \log(X))}\right)$$

where $Y = \mathrm{I}(\cdot)$ equals 1 if $U \leq F(X)$ and 0 otherwise.

1. Assuming one observes a sample of size n of the variables X and Y, comment the underlying model for that dataset while explaining its structure in plain English.

The indicator function constructs a test that compares a logistic regression function and uniform random variable. The logistic function returns a sample drawn from the uniform random variable X, and is compared to an independent sample of random variable from the uniform distribution, U. Where the logistic estimate is less than or equal to the uniform random variable, the indicator functions assigns a response of $Y = 1$, and 0 otherwise.

2. Construct an `R` function that generates a sample of size n of variables X and Y.

```
link <- function(b,n) {
  # draw two random samples of size n
  # from uniform distribution.
  #
  X <- runif(n)
  U <- runif(n)

  # The logit
  logit_X <- function(b,x) {
    (1 + exp(-b[1]-b[2]*x-b[3]*x^3-b[4]*log(x)))^-1
  }
```

```r
  # classify Y following classification scheme
  Y <- ifelse(U<=logit_X(b,X),1,0)
  return(data.frame(X=X,U=U,Y=Y))
}
```

3. Construct a box and whisker plot of test error rates.

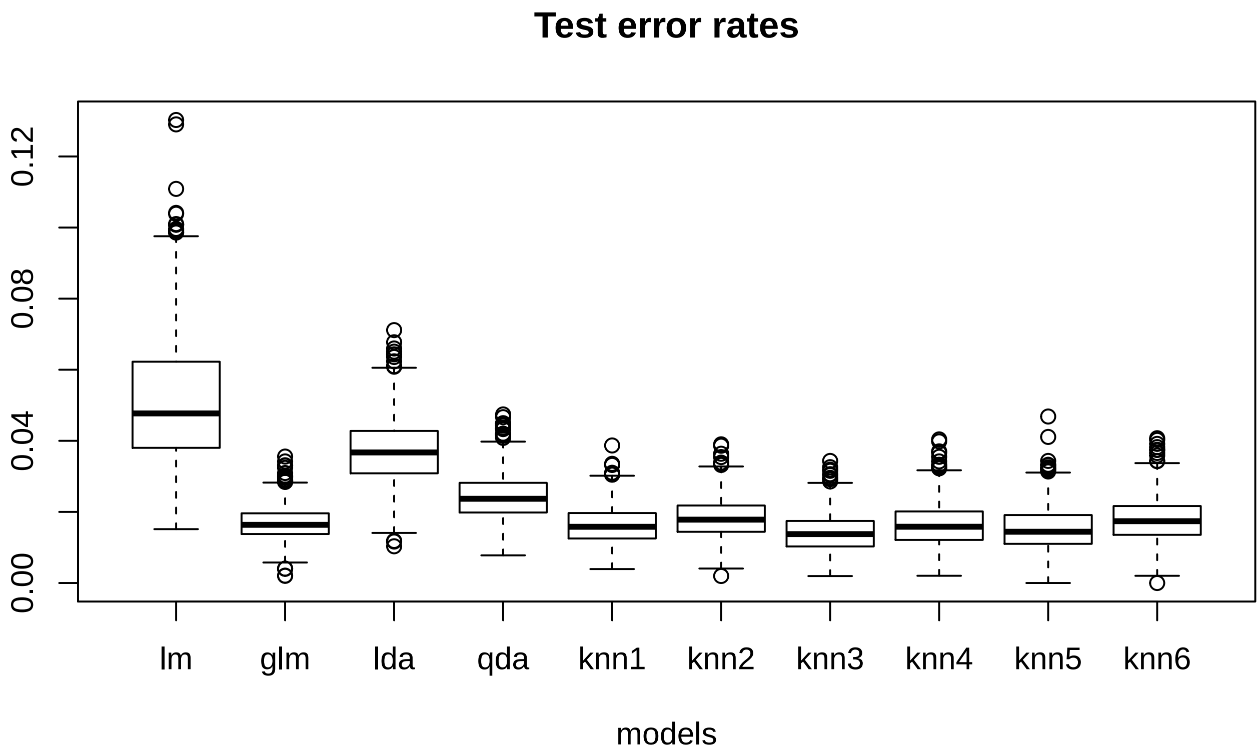
```r
# Helper function to draw samples
r.sample <- function(n,p=1/2) {
  train <- runif(n)<=p

  train
}

# Convenience wrapper to extract response from predict()
r.pred <- function(model,thr=0.5,newdata=NULL) {
  #' @model An object of class "lm","glm","lda","qda".
  #' @thr A scale of type `double`. `thr` sets the class threshold.
  #' @newdata An optional vector of new data.

  if(is.null(newdata)) print("No data")
  prob <- suppressWarnings(predict(model,newdata,type="response"))

  if (class(model)%in%c("lda","qda")) {
    pred <- prob$class

    return(
      pred
    )

  } else {
    pred <- rep(0,length(prob))
    pred[prob>thr]=1

    return(
      #vector of predictions
      pred
    )
  }
}
```

The figure below plots the distribution of test error rates observed from the bootstrap simulation of 10 classifying models.

```r
ter <- NULL
b <- c(-4,2,5,4) # beta constants
B <- 1000 # bootstraps
n <- 1000 # sample size
p <- 1/2 # hold out half

# draw sample from model
#
r <- link(b,n)

# the bootstrap
```

```r
#
for(i in 1:B) {
  #sampler
  # hold out strategy
  train <- r.sample(n,p)
  test <- !train

  # linear probability model
  #
  r.lm <- lm(Y~.,r,subset=train)
  r.lm.pred <- r.pred(r.lm,newdata=r[test,])
  ter_ <- data.frame(lm=mean(r.lm.pred!=r$Y[test]))
  # test error rate

  # logistic regression
  #
  r.glm <- glm(Y~.,r,family = binomial,subset=train)
  r.glm.pred <- r.pred(r.glm,newdata=r[test,])
  ter_$glm <- mean(r.glm.pred!=r$Y[test])

  # linear discriminant analysis
  #
  r.lda <- MASS::lda(Y~.,r,subset=train)
  r.lda.pred <- r.pred(r.lda,newdata=r[test,])
  ter_$lda <- mean(r.lda.pred!=r$Y[test])

  # quadratic discriminant analysis
  #
  r.qda <- MASS::qda(Y~.,r,subset=train)
  r.qda.pred <- r.pred(r.qda,newdata=r[test,])
  ter_$qda <- mean(r.qda.pred!=r$Y[test])

  # K-nearest neighbours
  #
  r.knn1 <- class::knn(
    use.all = TRUE,
    train=r[train,1:2],
    test=r[test,1:2],
    cl=r$Y[train],
    k=1
  )
  ter_$knn1 <- mean(r.knn1!=r$Y[test])

  r.knn2 <- class::knn(
    use.all = TRUE,
    train=r[train,1:2],
    test=r[test,1:2],
    cl=r$Y[train],
    k=2
  )
  ter_$knn2 <- mean(r.knn2!=r$Y[test])

  r.knn3 <- class::knn(
```

```
    use.all = TRUE,
    train=r[train,1:2],
    test=r[test,1:2],
    cl=r$Y[train],
    k=3
  )
  ter_$knn3 <- mean(r.knn3!=r$Y[test])

  r.knn4 <- class::knn(
    use.all = TRUE,
    train=r[train,1:2],
    test=r[test,1:2],
    cl=r$Y[train],
    k=4
  )
  ter_$knn4 <- mean(r.knn4!=r$Y[test])

  r.knn5 <- class::knn(
    use.all = TRUE,
    train=r[train,1:2],
    test=r[test,1:2],
    cl=r$Y[train],
    k=5
  )
  ter_$knn5 <- mean(r.knn5!=r$Y[test])

  r.knn6 <- class::knn(
    use.all = TRUE,
    train=r[train,1:2],
    test=r[test,1:2],
    cl=r$Y[train],
    k=6
  )
  ter_$knn6 <- mean(r.knn6!=r$Y[test])

  ter <- rbind(ter,ter_)
}

# plot figure box and whisker
boxplot(ter, xlab="models", main="Test error rates")
```



All models show a negative skew in the sampled distribution of test error rates. The linear probability model (`lm`) shows the greatest IQR of test error rates when compared to the set of models ($\pm 2.8\%$), and largest median test error rate (7.4%).

The logistic regression models (`glm`) show the smallest median test error rate (1.55%), which is to be expected given the form of the true model. Based on the observed test error rates, the author concludes that a logistic regression model presents the best predictor of Y.

The nearest neighbour classifier models (`knn`) are relatively similar when comparing test error rates IQR and median. $K = 1$ presents the lowest median test error rate (2.0%), and smallests observed IQR of all nearest neighbour classifier models (0.83%).

The quadratic descriminent model (`qda`) presents a marginal improvement to the linear descriminent model (`lda`) when comparing the variance and median levels of test error rates. The median test errors were 3.0% and 2.27% respectively.