MATH3346: Data Mining Honours - Assignment 2

Kevin Tan
u3207990

September 26, 2009

Question 1

Here we peform a simple test of the function `confusion()` using `lda()` to do linear discriminant calculations. We shall perform the test on the `Pima.tr` data (provided by the `MASS` library). The predictions are made using the function `lda()` applied to the `Pima.tr` class `'type'` versus all other columns as explanatory variables. Next, the function `confusion()` calculates the confusion matrix and evaluates the overall accuracy. Below, we obtain the confusion matrix (showing the class memberships) and the overall accuracy (approximately 75.5%):

```
> library(MASS)
>
> #obtain a prediction from using linear discriminant analysis
> PimaCV.lda <- lda(type~., data=Pima.tr, CV=TRUE)
>
> #check this model prediction against the actual original data
> confusion(Pima.tr$type, PimaCV.lda$class)
   Overall accuracy  Prior frequency.No Prior frequency.Yes
              0.755               0.660               0.340

Confusion matrix
      Predicted (cv)
Actual     No    Yes
   No  0.8636 0.1364
   Yes 0.4559 0.5441
```

The details of the help page of the function `confusion()` are filled in below:

```
\name{confusion}
\Rdversion{1.1}
\alias{confusion}
\title{Confusion Matrix}
\description{Calculates the confusion matrix based upon the data from
the actual classes versus the predicted classes (based on some model),
 and finds the overall predictive accuracy.}
\usage{confusion(actual, predicted, names = NULL, printit = TRUE,
prior = NULL)}
```

```
\arguments{
  \item{actual}{Actual classes from the data}
  \item{predicted}{Predicted classes from the model}
  \item{names}{User defined names for each of the classes. Defaults to
 class names provided in the 'actual' data.}
  \item{printit}{Whether to print the results to screen. Defaults to
TRUE.}
  \item{prior}{User defined prior probabilities for each class,
provided as a vector. Defaults prior probabilities to the proportions
of each class in the 'actual' data.}
}
\details{The function tries to calculate the confusion matrix based
upon the data from the actual classes versus the predicted classes.
The function allows prior probabilities to be user defined or
otherwise defaulted to the proportions of the data from the actual
classes. It also allows class names to be predefined or otherwise
defaulted to data from the 'actual' class names. In addition, the
function can be forced to not print on screen, otherwise it defaults
to printing out on the screen.

The confusion matrix is useful to test how a model predicts data
against the actual data. The function also provides the overall
accuracy based on the confusion matrix (calculated by the total sum of
 multiplying the prior probabilities of each class along the
respective diagonal of the confusion matrix).

The rows in the confusion matrix represent the actual classes, while
columns represent the predicted classes. Each of the probabilities in
the confusion matrix can show where the there is misclassification
according to specific classes. Along the main diagonal are the
probabilities of correct classification, while the off-diagonal are
the probabilities of incorrect classification.
}
\value{
The 'confusion' function returns a list of 3 components:

   \item{accuracy}{The overall accuracy of the calculated from the
confusion matrix.}
   \item{confusion}{The confusion matrix (based on the 'actual' data
versus the 'predicted' data, and any user specified prior
probabilities).}
   \item{prior}{The prior probabilities of each class (either user
defined or obtained from the proportions of each class in the 'actual'
 data).}
}
\references{
Maindonald, J. (2009) Statistical Perspectives on Data Mining.
Lectures notes for the ANU course MATH3346.

Maindonald, J. (2009) Exercises that Practice and Extend Skills in R.
Lab notes for the ANU course MATH3346.
http://en.wikipedia.org/wiki/Confusion_matrix
}
\author{
Maindonald, J. (wrote original confusion matrix code)
Tan, K. (wrote this help page)
}
\examples{
library(MASS)

#obtain a prediction using linear discriminant analysis
PimaCV.lda <- lda(type~., data=Pima.tr, CV=TRUE)

#check this model prediction against the actual original data
confusion(Pima.tr$type, PimaCV.lda$class)
}
```

Question 2

It is preferable to work with the logarithmic transformation of `re75`, rather than `re75`. This is due to the problem of the distribution of `re75` which has heavy tails and focuses too much on a small number of large values. This problem is avoided if we use `logre75`, which is the logarithmic transformation of `re75`.

(Also note that, in defining `logre75`, we add a small offset to `re75` before taking the logarithm, because the logarithm of zero is undefined. This offset should be half the non-zero minimum of `re75`.)

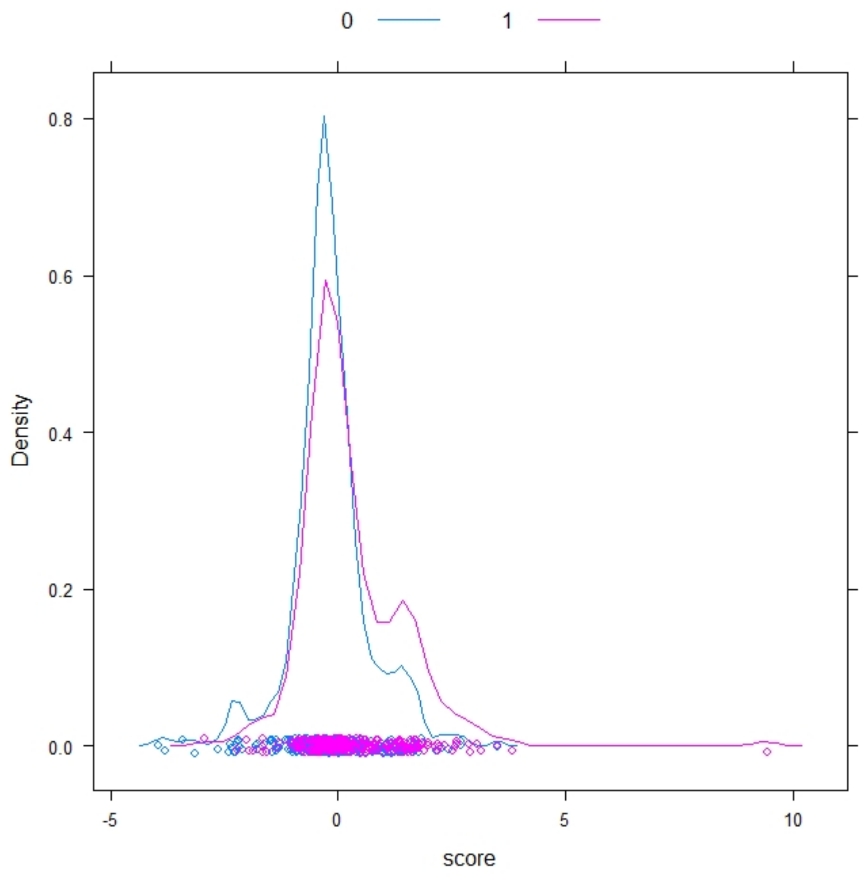
Question 3a

Here we run the function `check.model()` on `form1`, `form2` and `form3`

```
> check.model(form1)
[1] "Model is trt ~ age + educ + black + hisp + marr + nodeg + logre75"
Accuracy
  0.5679
> check.model(form2)
[1] "Model is trt ~ ns(age, 3) + educ + black + hisp + marr + nodeg + logre75"
Accuracy
  0.5748
> check.model(form3)
[1] "Model is trt ~ (ns(age, 3) + logre75) * (educ + black + hisp + marr) + "
[2] "Model is     nodeg"
Accuracy
  0.6025
```

We can see from above, the accuracies for `form1`, `form2` and `form3` are 56.8%, 57.5% and 60.0% respectively.

Since the `form3` model has the best accuracy of 60.0%, we shall used this model and plot its discriminant scores. The linear discriminant analysis (LDA) is applied against the `trt` term, where the `trt` term represents two groups. So for the plot below, the '0' curve represents the control group, and '1' curve represents the treatment group. (Note, that using LDA on only two groups produces only one set of discriminant scores.)



Given that the subjects were randomly divided between treatment and control groups, the plot above of the discriminant scores reflects this to a good degree. If the randomisation of assigning subjects into the two groups was done properly, then the two plots should be strongly identical.

Overall, the total densities are approximately the same, as seen by eye. The two plots above share similar peaks at the same scores. Along the scoring axis, at about score '0', the control group peak is slightly higher in density, while at about score '2', the treatment group peak is slightly higher in density. So these plots reflect what we expect from randomisation of subjects assigned to each group.

(We note that there is a curious outlier in the treatment group at about score '10', which stretches the plot.)

Question 3b

We shall compare the accuracies using `lda()`, from the previous question, with the accuracy by assigning all observations to the most frequent category. We shall do this by finding the root node accuracy using the function `rpart()`. Below we run the function `rpart()` on `form1`, `form2` and `form3`.

```
> library(rpart)
> rpart(form1, data=nswdem, method="class")
n= 722

node), split, n, loss, yval, (yprob)
      * denotes terminal node

 1) root 722 297 0 (0.5886427 0.4113573)
   2) educ< 11.5 563 217 0 (0.6145648 0.3854352)
     4) logre75< 7.104555 299 104 0 (0.6521739 0.3478261) *
     5) logre75>=7.104555 264 113 0 (0.5719697 0.4280303)
      10) logre75>=8.927776 70  23 0 (0.6714286 0.3285714)
        20) hisp< 0.5 63  18 0 (0.7142857 0.2857143) *
        21) hisp>=0.5 7   2 1 (0.2857143 0.7142857) *
      11) logre75< 8.927776 194  90 0 (0.5360825 0.4639175)
        22) logre75< 8.443862 151  63 0 (0.5827815 0.4172185)
          44) logre75>=8.323172 13   2 0 (0.8461538 0.1538462) *
          45) logre75< 8.323172 138  61 0 (0.5579710 0.4420290)
            90) age< 25.5 107  42 0 (0.6074766 0.3925234) *
            91) age>=25.5 31  12 1 (0.3870968 0.6129032) *
        23) logre75>=8.443862 43  16 1 (0.3720930 0.6279070)
          46) educ< 8.5 7   2 0 (0.7142857 0.2857143) *
          47) educ>=8.5 36  11 1 (0.3055556 0.6944444) *
   3) educ>=11.5 159  79 1 (0.4968553 0.5031447)
     6) age>=25.5 76  29 0 (0.6184211 0.3815789)
      12) age< 39.5 69  23 0 (0.6666667 0.3333333) *
      13) age>=39.5 7   1 1 (0.1428571 0.8571429) *
     7) age< 25.5 83  32 1 (0.3855422 0.6144578) *
> rpart(form2, data=nswdem, method="class")
n= 722

node), split, n, loss, yval, (yprob)
      * denotes terminal node

 1) root 722 297 0 (0.5886427 0.4113573)
   2) educ< 11.5 563 217 0 (0.6145648 0.3854352)
     4) logre75< 7.104555 299 104 0 (0.6521739 0.3478261) *
     5) logre75>=7.104555 264 113 0 (0.5719697 0.4280303)
      10) logre75>=8.927776 70  23 0 (0.6714286 0.3285714)
        20) hisp< 0.5 63  18 0 (0.7142857 0.2857143) *
        21) hisp>=0.5 7   2 1 (0.2857143 0.7142857) *
      11) logre75< 8.927776 194  90 0 (0.5360825 0.4639175)
        22) logre75< 8.443862 151  63 0 (0.5827815 0.4172185)
          44) logre75>=8.323172 13   2 0 (0.8461538 0.1538462) *
          45) logre75< 8.323172 138  61 0 (0.5579710 0.4420290)
            90) ns(age, 3).1< 0.03716701 107  42 0 (0.6074766 0.3925234) *
            91) ns(age, 3).1>=0.03716701 31  12 1 (0.3870968 0.6129032) *
        23) logre75>=8.443862 43  16 1 (0.3720930 0.6279070)
          46) educ< 8.5 7   2 0 (0.7142857 0.2857143) *
          47) educ>=8.5 36  11 1 (0.3055556 0.6944444) *
   3) educ>=11.5 159  79 1 (0.4968553 0.5031447)
     6) ns(age, 3).3>=-0.3256595 99  40 0 (0.5959596 0.4040404)
      12) ns(age, 3).3< 0.1290044 92  34 0 (0.6304348 0.3695652) *
      13) ns(age, 3).3>=0.1290044 7   1 1 (0.1428571 0.8571429) *
     7) ns(age, 3).3< -0.3256595 60  20 1 (0.3333333 0.6666667) *
> rpart(form3, data=nswdem, method="class")
Error in rpart(form3, data = nswdem, method = "class") :
  Trees cannot handle interaction terms
```

We can see from the above that the root node accuracies (by looking at the "`1) root`" level) are 58.9% for both `form1` and `form2` (we note that `rpart()` cannot be applied to `form3`, so its accuracy is unknown). This compares to the previous question, where using the function `lda()`, both `form1` and `form2` had accuracies of 56.8% and 57.5% respectively.

Linear discriminant analysis (using the function `lda()`) and decision trees (using `rpart()`) are two well-understood methodologies, which both make them suitable for analysis. LDA is more of a parametric method, while decision trees are highly non-parametric. It is useful to compare these two different methodologies, because LDA can have restrictive linearity assumptions. It can be complicated to model non-linear effects (such as in `form2` with respect to age term) and even interaction effects (such as in `form3`). By applying the methodology of decision trees, we can obtain a fairer comparison of the accuracies of the various models against LDA.

The best accuracies (as percentages) using LDA and decisions trees, as seen above, are actually relatively close. This suggests that the accuracies are probably a good measure of the models.

Finally, we shall comment on the randomisation, ie. if the sampling of the control and treatment groups are "truly" random. Based on 60% accuracy, we could say that the control and treatment groups (before training) are approximately similar. Yet based on about 40% error, we cannot say they are entirely similar. Thus randomisation could be suspect because the two groups should have stronger similarity (ie. higher accuracy from the models).

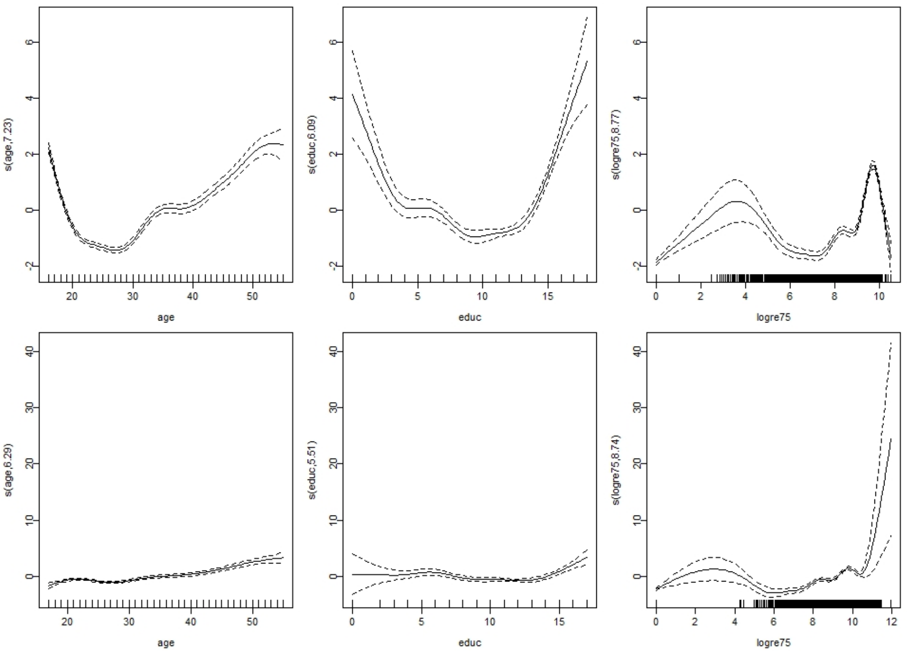
(If there was a higher percentage of accuracy, perhaps this would erase this suspicion. In drawing scientific analysis if training assists in earnings, it would be questionable, because the two groups are not obviously similar before training.)

Question 4

The number of rows for the `nswdem`, `cps1` and `psid1` data are 722, 15992 and 2490 respectively.

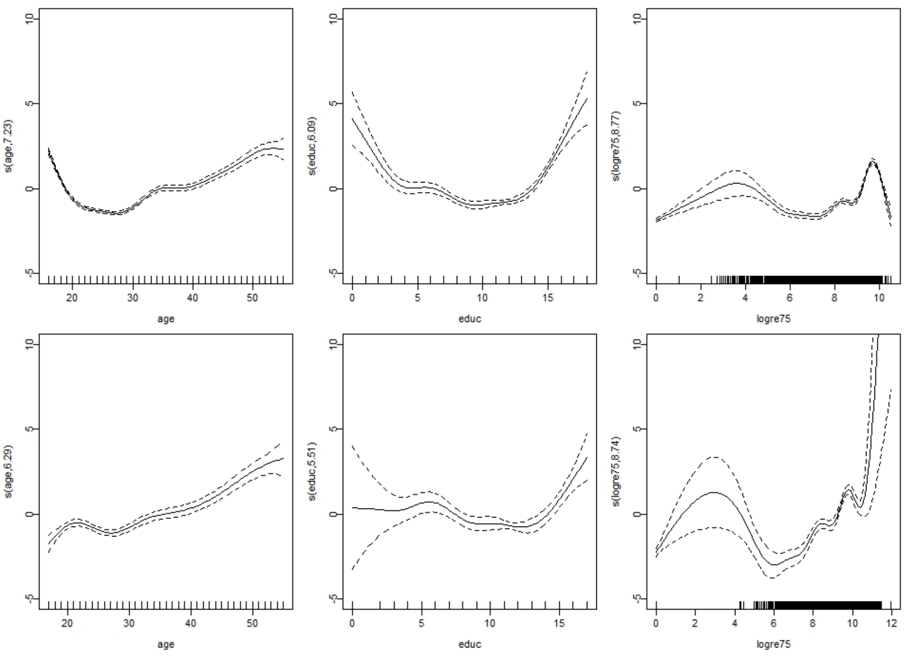
Here we shall use the function `gam()`, from the `mgcv` package. Generalised additive models (GAMs) apply generalised linear modelling with additive models, which can model non-linear effects (or residuals). In our particular case of using the function `gam()` below, the additive models shall use regression spline terms to model these non-linear effects.

After running the function `gam()` on both `cps1` and `psid1` data, we shall plot the non-linear effects as splines below. (We also note that in the code provided for the assignment, it is necessary to add the parameter `"family=binomial"` to the function `gam()`. This is because there exists categorical data, which requires logistic regression instead, ie. requiring the logit scale).



The top row of three plots show the `cps1` data versus the experimental (`nswdemo`) data, while the bottom row of three plots show the `psid1` data versus the experimental (`nswdemo`) data.

To check if transformations are effective for `cps1` data and experimental data compared `psid1` data and experimental data, we should check similarity of the columns of plots, as seen for `age` (age in years), `educ` (number of years education) and `logre75` ((logarithm of) real earnings in 1975). From the above plots, note that for the `logre75` plot for the `psid1` data, there is large scaling of the plot, due to the extreme spline fit at the right-side end of the plot. Due to this, the plots are not easily comparable by the eye. So instead the plots are replotted below, with all y-axis having the same range.



From the plots above, comparing pairwise the `cps1` and `psid1` plots, each of the `age`, `educ` and `logre75` plots have fitted splines for the non-linear effects which are generally quite similar in shape, scale and features (except as we noted before that the `logre75` plot has an extended x-axis scale for `psid1` than `cps1`.) This would suggest that the same transformations for the comparison between the experimental data and the `cps1` data are useful for the comparison between the experimental data and the `psid1` data.

Question 5

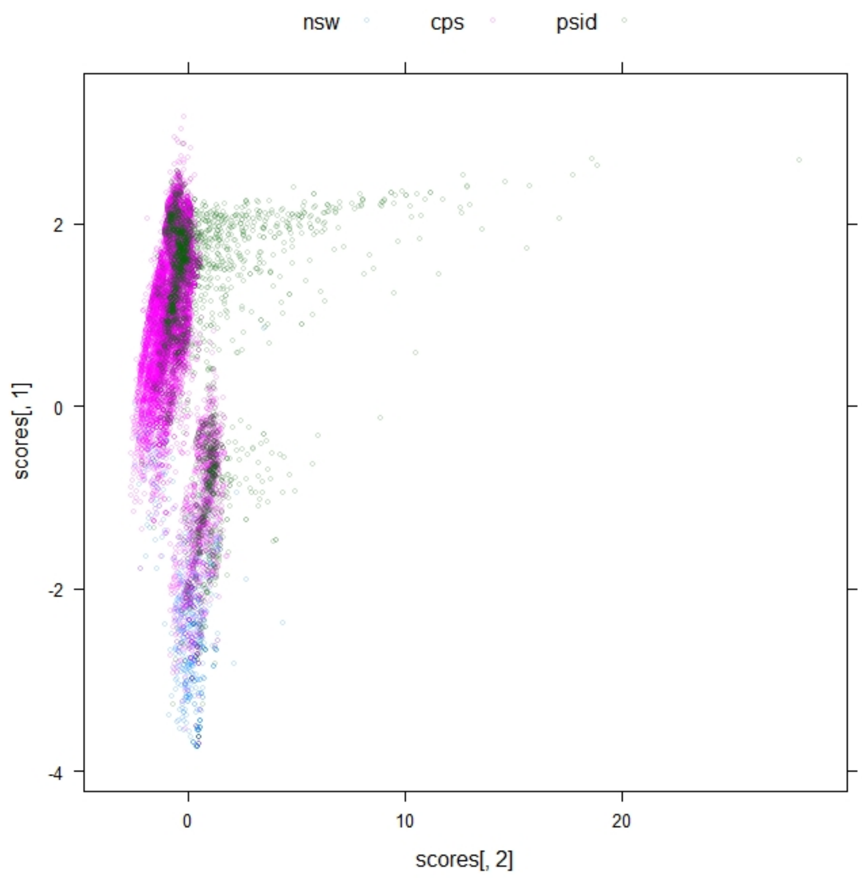
The function `lda()` is applied to the three groups, `nswdem`, `cps1` and `psid1`. Two models are tested using `lda()`, where the first model is a simple linear model, while the second model uses splines, because question 4 suggests that modelling non-linear effects can be helpful. (We note that splines of only degree 4 are used, because care must be taken to not overfit the data.)

Below the two models are tested using the function `check.model()`, which also has an additional parameter "`df=nswplus`" because we wish to use the dataframe `nswplus` incorporating all three groups.

```
> check.model(fm1, df=nswplus)
[1] "Model is gp ~ age + educ + black + hisp + marr + nodeg + logre75"
Accuracy
  0.8222
> check.model(fm2, df=nswplus)
[1] "Model is gp ~ ns(age, 4) + ns(educ, 4) + ns(logre75, 4) + black + hisp + "
[2] "Model is     marr + nodeg"
Accuracy
  0.8496
```

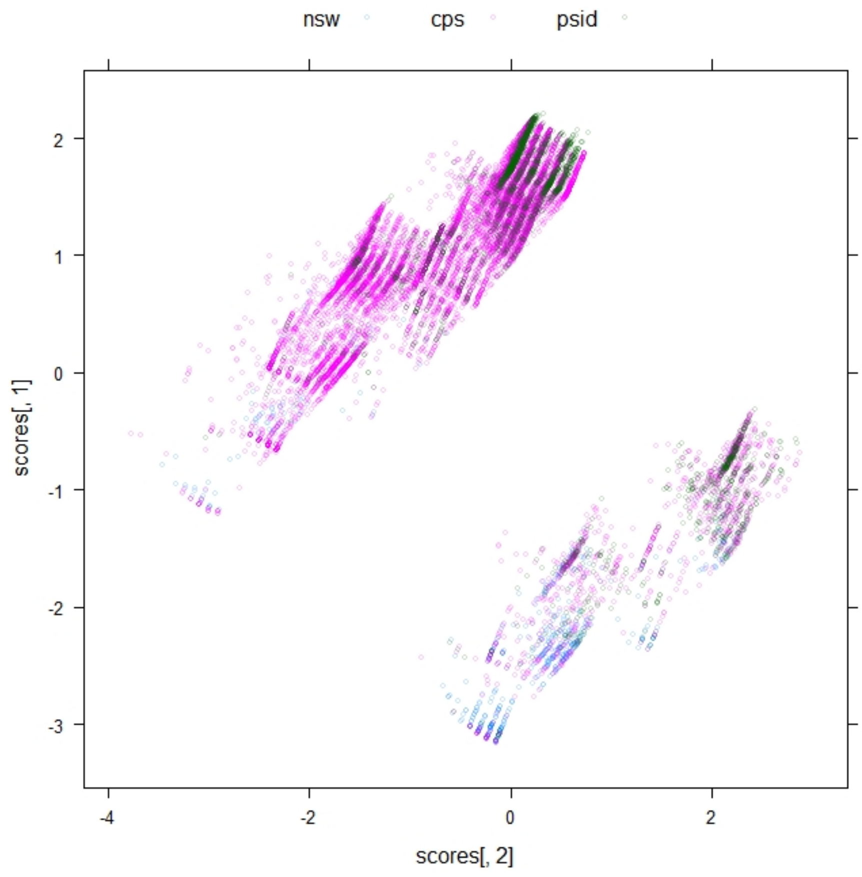
The first simple linear model has an accuracy of 82.2%, while the second model using splines has a higher accuracy of 85.5%.

Since the second model "fm2" has a higher accuracy, we shall plot the fit from the linear discriminant analysis. Using LDA on three groups, the two-dimensional plot is the complete summary of the LDA fit.



The 2D discriminant scoring above shows the `nswdemo` data as blue points, `cps1` data as pink points, and the `psid1` data as green points. The striking aspect of this plot is the spread of the `psid1` points from the top-left to the right. In fact looking at the splits of the data, this spread of this range is due to the `logre75` term being modelled as a spline. Looking back at question 4, we noted in the spline fitting for `logre75`, the `psid1` plot had an extended x-axis range and a spline fit with an extreme high end, as compared to the `cps1` plot. This suggests although the accuracy percentage of fitting a model with splines has a greater accuracy than fitting a simple linear model, it can be misleading that this model fit may not be well represented as 2D discriminant plot, which questions the credibility of the spline model.

Below, we shall also plot the linear model "`fm1`":



For this particular plot above compared to the previous plot, we can see that `psid1` does not have the extended spread of points. Both plots also share a striking feature, that both have two distinct clusters. The variables were examined to find the cause. This was done by testing the removal of only one explanatory variable from the model. After testing against every singularly removed explanatory variable, we find that removing the `black` explanatory variable from both models show plots of only one cluster. Thus the two clusters are divided by the `black` variable, where the upper cluster is `black=0` (person is not black), while the lower cluster is `black=1` (person is black).

Question 6a

Here we annotate the random forest code below, by adding comments to explain what each step does and the choice of function arguments:

```
> # Here we must load the 'randomForest' package first, in order to
use the function randomForest() later.
> library(randomForest)

> ## Extract subset of data for plotting
> # We know that 'nswplus' is the combined datasets of 'nswdem', 'cps1
' and 'psid1'.
```

```
> # The 'm' vector has the initial index values where each dataset is
inside 'nswplus'.
> # The 'm' vector is calculated from the cumulative sum on the number
 of rows of each dataset.
> m <- cumsum(c(nrow(nswdem), nrow(cps1), nrow(psid1)))

> # Here a random sample of 500 integers (without replacement) are
taken inside the range of the indices of each dataset for 'nswplus'.
> n1 <- sample(1:m[1], 500)
> n2 <- sample((m[1]+1):m[2], 500)
> n3 <- sample((m[2]+1):m[3], 500)

> # The random samples are concatenated into one vector. This
represents the sample that shall be used for prediction later. The
values inside 'take' are actually the indices that shall be sampled
from 'nswplus'.
> take <- c(n1,n2,n3)

> # Random forest is used for prediction.
> # We shall use the simple linear model 'fm1'.
> # 'sampsize=rep(722,3)' means that 722 elements are sampled from
each of the 3 datasets/classes (firstly, the sample size is made the
size of the 'nswdemo' experimental groups, and secondly, we wish to
keep sample size equal for each group, so the prior probabilities for
each dataset are equal).
> nswplus.rf <- randomForest(fm1, data=nswplus, sampsize=rep(722,3))

> # Here the random forest model from above is tested on the random
sample using 'take'. It tries to make the predictions of each of the
samples
> nswplus.pred <- predict(nswplus.rf, newdata=nswplus[take, ],
proximity=TRUE)
> The "distance" matrix is found from 'nswplus.pred$proximity'. The '
proximity' values mean that towards 0 is farther away, while towards 1
 is closer. So the 'proximity' values are subtracted from 1, in order
to get the "distance" matrix where a larger number means further away.
> sim <- 1-nswplus.pred$proximity

> # The function cmdscale() is "classical multidimensional scaling of
a data matrix".
> # This essentially converts the "distance" matrix into a low-
dimensional representation of the points, while trying to preserve the
 inter-distances between the points. (Note, the function cmdscale()
defaults to two dimensions.)
> scores.rf <- cmdscale(sim)

> # This creates a 2D ordination plot of the sampled points from the
random forest prediction.
> # A legend is produced (using 'auto.key') and the size of the points
 are changed (using 'par.settings').
```

```
> xyplot(scores.rf[,1] ~ scores.rf[,2], groups=nswplus$gp[take],
       auto.key=list(columns=3), par.settings=simpleTheme(cex=0.5,
alpha=0.5))
```

Question 6b

Based on the previous question 5, we should check if the "linear" model using LDA is adequate. A good comparison is against a highly non-parametric model, such as random forests (which can handle complex interactions of the explanatory variables almost automatically). Random forests are nearly self-automated, where it can handle explanatory variables and factors in relatively complex ways, which is an advantage over LDA. LDA has the downside that for increased predictive accuracy, it may require more complex forms of interactions of the explanatory variables.

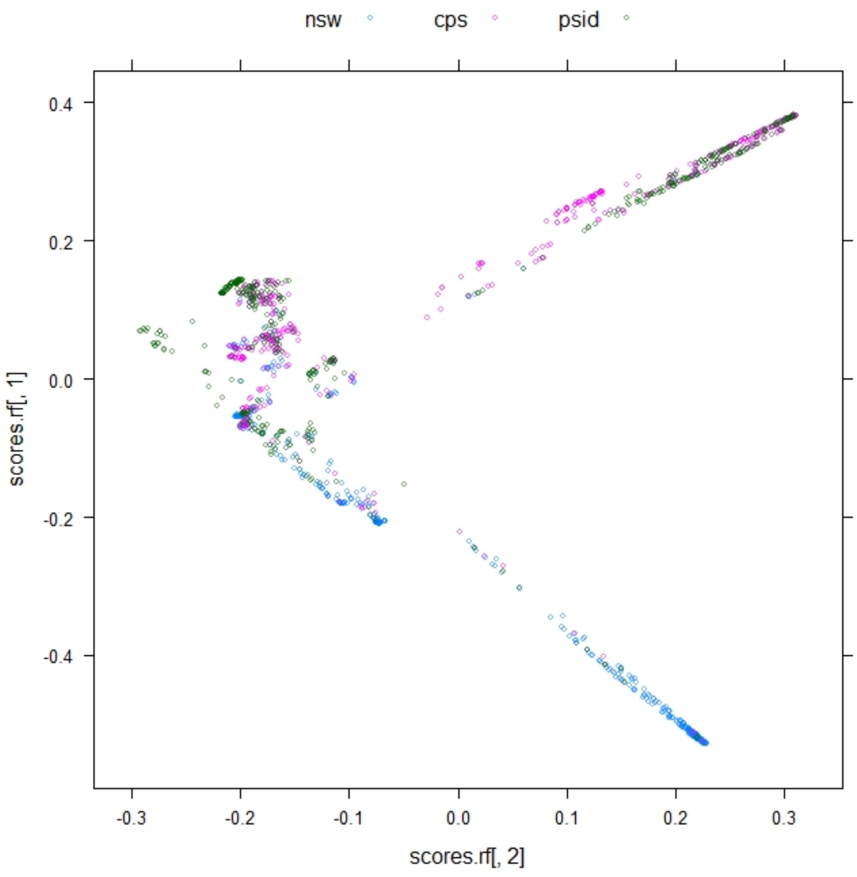
Here we shall find the out-of-bag (OOB) error rate for the random forest prediction, and the test set error rate using the 'take' sample (to check if there is overfitting). We also run the code a couple of times to get an idea of the accuracy:

```
> randomForest(fm1, data=nswplus, sampsize=rep(722,3), xtest=nswplus[
take, c(-1,-8,-9,-10,-11,-13)], ytest=nswplus[take,11])

Call:
 randomForest(formula = fm1, data = nswplus, sampsize = rep(722,
3), xtest = nswplus[take, c(-1, -8, -9, -10, -11, -13)],      ytest =
nswplus[take, 11])
               Type of random forest: classification
                     Number of trees: 500
No. of variables tried at each split: 2

        OOB estimate of  error rate: 20.91%
Confusion matrix:
     nsw   cps psid class.error
nsw  634    54   34   0.1218837
cps  840 13166 1986   0.1767134
psid 188   913 1389   0.4421687
                Test set error rate: 22.47%
Confusion matrix:
     nsw cps psid class.error
nsw  457  32   11       0.086
cps   29 416   55       0.168
psid  30 180  290       0.420

> randomForest(fm1, data=nswplus, sampsize=rep(722,3), xtest=nswplus[
take, c(-1,-8,-9,-10,-11,-13)], ytest=nswplus[take,11])

Call:
 randomForest(formula = fm1, data = nswplus, sampsize = rep(722,
3), xtest = nswplus[take, c(-1, -8, -9, -10, -11, -13)],      ytest =
nswplus[take, 11])
               Type of random forest: classification
                     Number of trees: 500
```

```
No. of variables tried at each split: 2

        OOB estimate of  error rate: 20.46%
Confusion matrix:
     nsw   cps psid class.error
nsw  628    55   39   0.1301939
cps  836 13292 1864   0.1688344
psid 191   945 1354   0.4562249
                Test set error rate: 22.8%
Confusion matrix:
     nsw cps psid class.error
nsw  457  31   12       0.086
cps   29 419   52       0.162
psid  35 183  282       0.436
```

We can see the OOB error rate (which is comparable to the cross-validation error rate for LDA) is approximately 20%, while the test-set error rate is about 23% (which are both similar in error, so the random forest prediction is not significantly overfitting). Using the empirical error rate of about 23%, we can see the random forest predictive accuracy is about 77%. Compared to the previous question 5 using LDA, we saw that using the same model `fm1`, it had a predictive accuracy of about 82%. Since the accuracies are not significantly different, and the accuracy percentages are high, then we could say based on this the groups are similar. However, before drawing a conclusion, we should examine the ordination plot.

The plot below shows ”distances” between the plots which represent the proportion of times the points down the tree that appear at the same terminal node.



The blue points are from `nswdemo` data, the pink points are from the `cps1` data, and the green points are from the `psid1` data.

Despite the high accuracy percentages showing strong predictive accuracy, the plot above shows that it cannot be entirely relied upon. We can see the `nswdemo` points have travelled down quite a separate path to the terminal node. Whereas, we can see the stronger relationship of the `cps1` and `psid1` points which are more overlapping. This would suggest that although the predictive accuracy is strong for LDA and random forests, it would mostly likely apply only to the `cps1` and `psid1` data, rather than the `nswdemo` data. Thus based on the plot above, the nswdemo experimental group compared to both `psid1` and `cps1` non-experimental control groups are probably inherently different. This brings about the question of the source and target issue. If the analysis on the `nswdemo` data were made, the results cannot be justify applied targeting the same source of people from the non-experimental control groups, `cps1` and `psid1`.

Question 7

Here we shall describe on what can be deduced from the several steps of the analysis:

Question 3 - This analysis to check if the control and treated groups if they divided properly by good randomisation. This was tested by using two well understood methodologies: a parametric approach, linear discriminant analysis, versus a non-parametric approach, decision trees. They both gave similar predicative accuracies, which meant that the accuracy percentage is probably a good indication. However, since the predictive accuracy was lower than expected, the randomisation of the groups is questionable. This could mean that the scientific results (ie. if training is beneficial for increased earnings), utilising these groups, would be possibly unreliable.

Question 4 - Using generalised additive models (GAMs) with spline methodologies, we showed that the same transformations were possibly effective for the comparison between the experimental data and `cps1` data were useful between the experimental data and `psid1` data. The additive models is considered here to see how modelling non-linear effects can be an improvement. From the plots, we showed that the non-linear effects of the `cps1` and `psid1` data were actually similar. This suggested that `cps1` data and `psid1` data had good similarity. However, there was problem with the extreme spline fit of large `logre75` in the `psid1` data, which showed up in the next question 5.

Question 5 - Predictive accuracy using linear discriminant analysis was tested on a simple linear model and the spline methodology (which seems promising based on question 4). The spline model had a greater margin of predictive accuracy. However, it was seen in the discriminant plot there was a problem with the spread of `logre75` in the `psid1` data. Also, we saw that the plots showed two distinct clusters which was due to the one being people being black and the other being non-black.

Question 6 - Random forests was tested to compare it to the linear discriminant analysis in question 5. It is useful to compare two quite different methodologies: linear discriminant analysis, as a parametric method, versus random forests, as a highly non-parametric method. Another benefit over linear discriminant analysis is that random forests automatically considers possibly more complex interactions of explanatory variables. In the end, the predictive accuracies were all quite strong for both random forests and linear discriminant analysis. However, in examining the ordination plot, it showed that the non-experimental groups `psid1` and `cps1` were similar, but the experimental group `nswdemo` was quite different.

Finally, this would question the reliability of the results derived from the experimental groups (as a source), in being applied (as a target) to the same source of the non-experimental control groups. This is a problem of source and target. The experiment probably took a sample from a quite a different source compared to the source of the non-experimental control sample.

Code Appendix:

```
##Setup
opt <- options(SweaveHooks=list(fig=function(){par(cex.main=1.0, cex
=0.8,
                        mar=c(4.1,3.6,1.6,1.1), pty="s", lwd=1,
                        mgp=c(2.25,0.5,0), tck=-0.025)}),
               prompt=" ", continue="  ")
pdf.options(pointsize=8)


##Useful functions
samprows <- function(df, m)df[sample(1:nrow(df), m), ]

confusion <- function(actual, predicted, names=NULL,
                     printit=TRUE, prior=NULL){
  if(is.null(names))names <- levels(actual)
  tab <- table(actual, predicted)
  acctab <- t(apply(tab, 1, function(x)x/sum(x)))
  dimnames(acctab) <- list(Actual=names,
                     "Predicted (cv)"=names)
  if(is.null(prior)){
    relnum <- table(actual)
    prior <- relnum/sum(relnum)
    acc <- sum(tab[row(tab)==col(tab)])/sum(tab)
  } else
  {
    acc <- sum(prior*diag(acctab))
    names(prior) <- names
  }
  if(printit)print(round(c("Overall accuracy"=acc,
                          "Prior frequency"=prior),4))
  if(printit){
    cat("\nConfusion matrix", "\n")
    print(round(acctab,4))
  }
  invisible(list(accuracy=acc, confusion=acctab, prior=prior))
}

check.model <- function(form, df=nswdem, dp=4, prior=NULL, discfun=lda
){
    categ <- all.names(form)[2]
    if(is.null(prior)){
        df.disc <- discfun(form, data=df, CV=TRUE)
        acc <- confusion(df[, categ], df.disc$class, printit=FALSE)
$accuracy
    } else {
       df.disc <- discfun(form, data=df, CV=TRUE, prior=prior)
       acc <- confusion(df[, categ], df.disc$class, printit=FALSE,
                       prior=prior)$accuracy
```

```
   }
    print(paste("Model is", deparse(form)))
    print(c(Accuracy = round(acc, dp)))
    invisible(acc)
}

##Datasets
#nswdem
library(DAAG); library(DAAGxtras)
library(splines)
dim(nswdemo)

library(MASS)
unique(sort(nswdemo$re75))[2]/2

nswdem <- nswdemo
nswdem$logre75 <- log(nswdemo$re75+37)

#nswplus
library(DAAG); library(DAAGxtras)
nswplus <- rbind(nswdemo, cps1, psid1)
nswplus$gp <- factor(rep(c("nsw","cps","psid"),
                        c(722, nrow(cps1), nrow(psid1))),
                    levels=c("nsw","cps","psid"))
nswplus$logre75 <- log(nswplus$re75+1)
nswplus$gp1 <- 1-as.numeric(nswplus$gp=="nsw")
  # gp1 distinguishes between the experimental and other data
## Now check that the new dataset has been correctly created
all.equal(nswplus[nswplus$gp=="nsw",1:10],nswdemo)
all.equal(nswplus[nswplus$gp=="cps",1:10],cps1, check.attributes=FALSE
)
all.equal(nswplus[nswplus$gp=="psid",1:10],psid1, check.attributes=
FALSE)
  # NB all.equal() checks for accuracy to within some small tolerance
  # With all.equal(), NAs in the same location count as "equality"

##QUESTION 1
prompt(confusion)

library(MASS)

#obtain a prediction using linear discriminant analysis
PimaCV.lda <- lda(type~., data=Pima.tr, CV=TRUE)

#check this model prediction against the actual original data
confusion(Pima.tr$type, PimaCV.lda$class)


##QUESTION 3
library(lattice)
```

```
form1 <- trt ~ age+educ+black+hisp+marr+nodeg+logre75
form2 <- trt ~ ns(age,3)+educ+black+hisp+marr+nodeg+logre75
  # form2 allows for the possibility that the effect of age may be
nonlinear
form3 <- trt ~ (ns(age,3)+logre75)*(educ+black+hisp+marr)+nodeg
  # form3 allows for interaction effects involving continuous
variables
## Try also the equivalent models with form2 and form3.
check.model(form1)
check.model(form2)
check.model(form3)

##QUESTION 3a
form <- form3
nswdem.lda <- lda(form, data=nswdem)
score <- predict(nswdem.lda)$x
plot(densityplot(~score, groups=nswdem$trt, auto.key=list(columns=2)))

##QUESTION 3b
library(rpart)
rpart(form1, data=nswdem, method="class")
rpart(form2, data=nswdem, method="class")
rpart(form3, data=nswdem, method="class")


##QUESTION 4
#find the number of rows for nswdem, cps1 and psid1
nrow(nswdem)
nrow(cps1)
nrow(psid1)

#create the GAM of the 'cps1' and 'psid1' data
library(mgcv)
form <- gp1 ~ s(age)+s(educ)+s(logre75)+black+hisp+marr+nodeg
nswcps.gam <- gam(form, data=subset(nswplus, gp!="psid"), family=
binomial)
nswpsid.gam <- gam(form, data=subset(nswplus, gp!="cps"), family=
binomial)
#plot the GAM of the 'cps1' and 'psid1' data
opar <- par(mfrow=c(2,3), mar=c(3.6,3.6,0.6,0.6), mgp=c(2.25,.5,0))
plot(nswcps.gam)
plot(nswpsid.gam)
par(opar)

#scale the GAM plots to the same y-axis scales, to make a fairer
comparsion
opar <- par(mfrow=c(2,3), mar=c(3.6,3.6,0.6,0.6), mgp=c(2.25,.5,0))
plot(nswcps.gam, ylim=c(-5,10))
plot(nswpsid.gam, ylim=c(-5,10))
```

```
par(opar)


##QUESTION 5
library(splines)
fm1 <- gp ~ age+educ+black+hisp+marr+nodeg+logre75
fm2 <- gp ~ ns(age,4)+ns(educ,4)+ns(logre75,4)+black+hisp+marr+nodeg
  # We prefer to give the three groups equal say

#check the predictive accuraacy using check.model()
check.model(fm1, df=nswplus)
check.model(fm2, df=nswplus)

#plot of the lda fit
nswplus.lda <- lda(fm2, data=nswplus, prior=rep(1,3)/3)
scores <- predict(nswplus.lda)$x
library(lattice)
xyplot(scores[,1] ~ scores[,2], groups=nswplus$gp,
       auto.key=list(columns=3), par.settings=simpleTheme(cex=0.5,
alpha=0.2))

##QUESTION 6a
library(randomForest)
# Extract subset of data for plotting
m <- cumsum(c(nrow(nswdem), nrow(cps1), nrow(psid1)))
n1 <- sample(1:m[1], 500)
n2 <- sample((m[1]+1):m[2], 500)
n3 <- sample((m[2]+1):m[3], 500)
take <- c(n1,n2,n3)
nswplus.rf <- randomForest(fm1, data=nswplus, sampsize=rep(722,3))
nswplus.pred <- predict(nswplus.rf, newdata=nswplus[take, ], proximity
=TRUE)
sim <- 1-nswplus.pred$proximity
scores.rf <- cmdscale(sim)
xyplot(scores.rf[,1] ~ scores.rf[,2], groups=nswplus$gp[take],
       auto.key=list(columns=3), par.settings=simpleTheme(cex=0.5,
alpha=0.5))
##QUESTION 6b
#check the out-of-bag error rate and test set error (based on the '
take' sample)
randomForest(fm1, data=nswplus, sampsize=rep(722,3), xtest=nswplus[
take, c(-1,-8,-9,-10,-11,-13)], ytest=nswplus[take,11])
#restore-defaults
options(opt)
```