Credit Scoring Coursework 1

Alexander John Pinches

23 November 2018

Preprocessing

We can use **summary** to show summarise the predictors and **str** to see their structure to help determine what we need to do to the data before training our model.

```
D1 <- D1[,c(1,2,4,5,10,33)]#remove unused predictors
summary(D1)#summarise dataframe
```

```
##   def_flag         loan_amnt         grade        emp_length_p
##  Mode :logical   Min.   : 1000   Min.   :1.00   Min.   : 0.000
##  FALSE:138375    1st Qu.: 8250   1st Qu.:2.00   1st Qu.: 3.000
##  TRUE :18710     Median :13000   Median :3.00   Median : 7.000
##                  Mean   :14877   Mean   :2.89   Mean   : 6.117
##                  3rd Qu.:20000   3rd Qu.:4.00   3rd Qu.:10.000
##                  Max.   :35000   Max.   :7.00   Max.   :10.000
##                                                 NA's   :8063
##       term         addr_state
##  Min.   :36.00   CA     :22261
##  1st Qu.:36.00   NY     :13247
##  Median :36.00   TX     :12491
##  Mean   :43.43   FL     :10493
##  3rd Qu.:60.00   IL     : 6470
##  Max.   :60.00   NJ     : 5874
##                  (Other):86249
```

```
str(D1)#show structure
```

```
## 'data.frame':    157085 obs. of  6 variables:
##  $ def_flag    : logi  TRUE FALSE FALSE FALSE FALSE FALSE ...
##  $ loan_amnt   : int  12000 32425 27000 18000 15000 17000 24000 8000 16000 4000 ...
##  $ grade       : num  1 3 2 3 4 2 4 2 5 6 ...
##  $ emp_length_p: num  NA 10 3 4 10 3 6 4 1 NA ...
##  $ term        : num  36 60 60 36 60 36 60 36 60 36 ...
##  $ addr_state  : Factor w/ 50 levels "","AK","AL","AR",..: 5 25 16 6 44 25 12 16 25 35 ...
```

We can see that term may only take certain values, **emp_length_p** contains NA's and we should investigate how **loan_amnt** is distributed.**addr_state** appears to be in an appropriate form and doesnt need preprocessing as it's already of type factor and contains no NA's. Grade is a numeric in the dataframe and should be a factor with 7 levels by it's definition.

```
unique(D1$term) #show unique values term takes
```

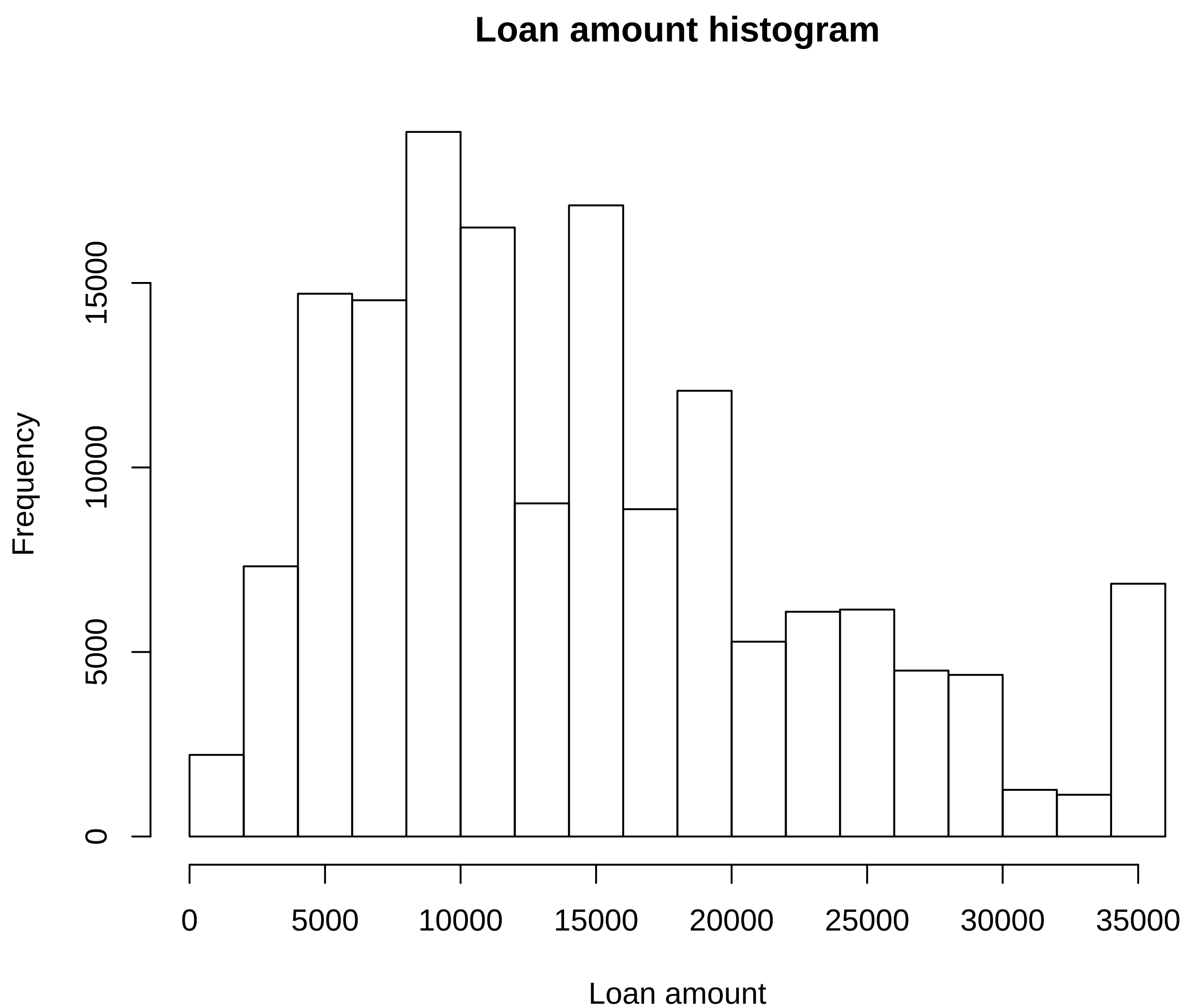
```
## [1] 36 60
```

```
D1 %>% group_by(emp_length_p) %>% tally() #show values it takes and ammount of each using dplyr package
```

```
## # A tibble: 12 x 2
##    emp_length_p     n
##           <dbl> <int>
##  1            0 12024
##  2            1  9751
##  3            2 13746
##  4            3 11996
##  5            4  8986
##  6            5  8624
##  7            6  7815
##  8            7  8724
```

```
## 9             8  7934
## 10            9  6392
## 11           10 53030
## 12           NA  8063
```

```r
hist(D1$loan_amnt, main="Loan amount histogram",
     xlab="Loan amount")#plot histogram of loan amount
```



Using `unique` we can see term only takes two values so may be more appropriate as a factor with 2 levels. Using `dplyr` we can see `emp_length_p` contains NA's looking at the explanation of this predictor it is unclear whether the NA's are because they have no job or are random. So as they make up a small proportion of the data set we will remove those rows containing NA's. The data is also heavily skewed towards 10 although there's no way to transform the data to remove this. Looking at the histogram of loan amount we see that there may be some extreme values to combat this we will take the log of all the values to transform the dataset's distribution. We can see from the description of grade should be of type factor. We will make these changes below.

```r
D1$grade <- as.factor(D1$grade) #set grade as factor
D1$term <- as.factor(D1$term) #set term as factor
D1$loan_amnt <- as.numeric(lapply(D1$loan_amnt,log)) #take log of loan_amnt
D1 <- D1[which(!is.na(D1$emp_length_p),arr.ind = T),] #remove rows with NA in emp_length_p
```

We can now check that the data set is now in the form we want and show the distribution of the log of the loan amounts.

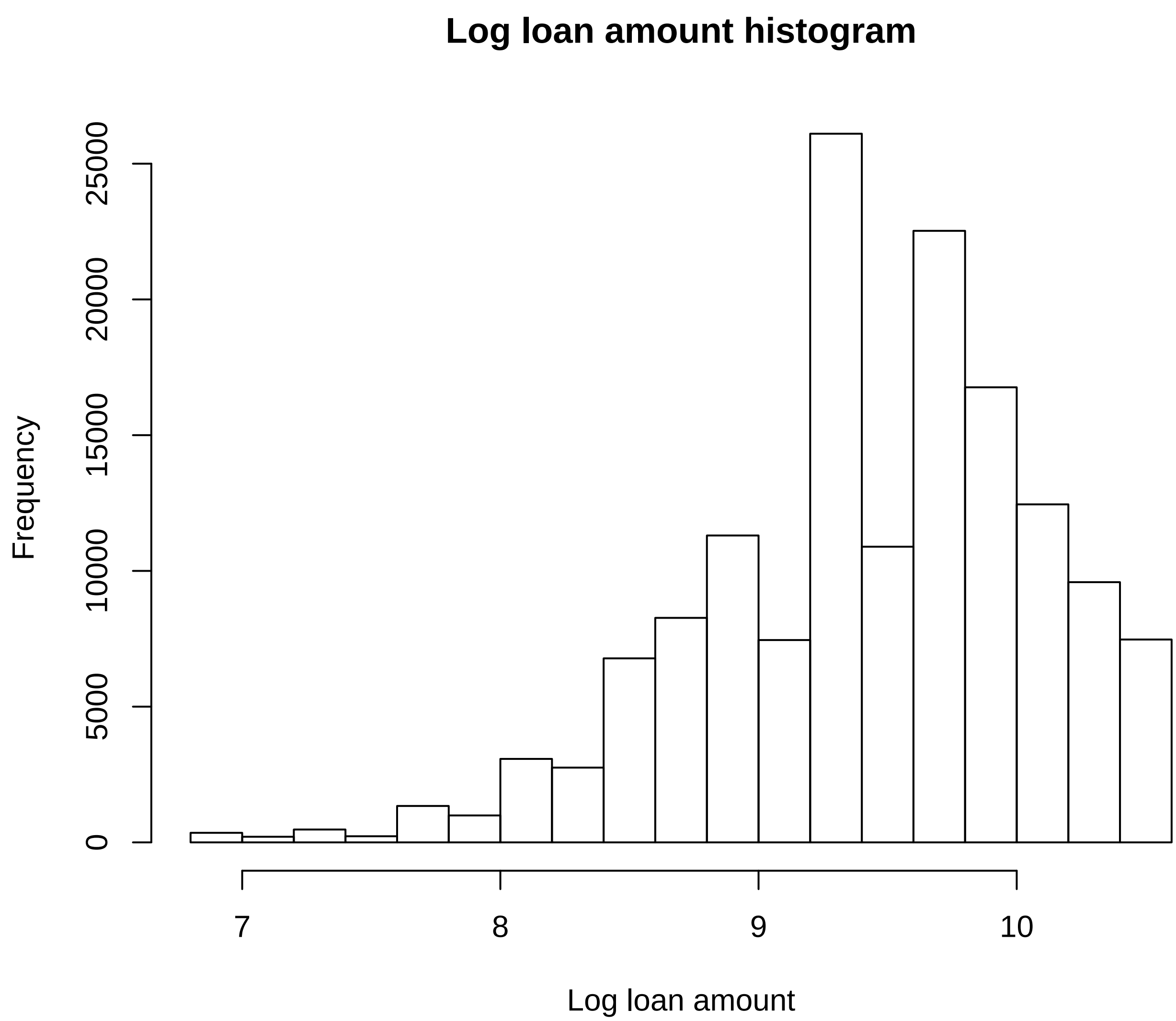
```r
summary(D1)#summarise dataframe
```

```
##   def_flag          loan_amnt        grade       emp_length_p     term
##  Mode :logical   Min.   : 6.908   1:23111   Min.   : 0.000   36:101955
##  FALSE:131556    1st Qu.: 9.048   2:39382   1st Qu.: 3.000   60: 47067
##  TRUE :17466     Median : 9.510   3:41851   Median : 7.000
##                  Mean   : 9.437   4:27077   Mean   : 6.117
##                  3rd Qu.: 9.903   5:12580   3rd Qu.:10.000
##                  Max.   :10.463   6: 3914   Max.   :10.000
##                                   7: 1107
##    addr_state
##  CA      :21223
##  NY      :12573
##  TX      :12003
##  FL      : 9829
##  IL      : 6168
##  NJ      : 5638
##  (Other):81588
```

```
str(D1)#show structure
```

```
## 'data.frame':    149022 obs. of  6 variables:
##  $ def_flag    : logi  FALSE FALSE FALSE FALSE FALSE FALSE ...
##  $ loan_amnt   : num  10.39 10.2 9.8 9.62 9.74 ...
##  $ grade       : Factor w/ 7 levels "1","2","3","4",..: 3 2 3 4 2 4 2 5 2 3 ...
##  $ emp_length_p: num  10 3 4 10 3 6 4 1 8 7 ...
##  $ term        : Factor w/ 2 levels "36","60": 2 2 1 2 1 2 1 2 2 2 ...
##  $ addr_state  : Factor w/ 50 levels "","AK","AL","AR",..: 25 16 6 44 25 12 16 25 43 47 ...
```

```
hist(D1$loan_amnt, main="Log loan amount histogram",
     xlab="Log loan amount")#plot histogram of log loan amount
```



The data is in the intended forms and we have transformed loan amounts to a better distribution as shown in the histogram above.

Creating test and training data

We can take a random sample without replacement from the rows of the dataset of size 2/3 of the whole dataset. We then use the left over rows as the test set.

```
sample <- sample(nrow(D1), size = 2*nrow(D1)/3)#create sample indicies
outofbag <- setdiff(1:nrow(D1), sample)#calculate remaining indicies
train <- D1[sample,]#make training data
test <- D1[outofbag,]#make testing data
```

Creating scorecard

We first create a single logistic regression model using the predictors and the training dataset with the above transformations having been performed.Not including interaction terms.

```
model <- glm(def_flag~loan_amnt+term+grade+emp_length_p+addr_state,
             family = binomial(link='logit'),data = train)#train logit model
```

From this model we can extract all the components of a scorecard namley the coefficents and their p-values by using `coef` and `summary` along with their standard error and z value and then save it in memory.

```r
scorecard <- coef(summary(model))#make score card from model
scorecard#print score card
```

```
##                  Estimate  Std. Error      z value      Pr(>|z|)
## (Intercept)   -4.68931626  0.29762512 -15.75578131  6.267990e-56
## loan_amnt      0.09680365  0.01740871   5.56064567  2.687784e-08
## term60        -0.16789871  0.02566329  -6.54236837  6.055213e-11
## grade2         0.79513073  0.04953358  16.05235633  5.503436e-58
## grade3         1.38150884  0.04785791  28.86688381 3.110859e-183
## grade4         1.79186517  0.04933817  36.31803117 8.402552e-289
## grade5         2.12105217  0.05307258  39.96512205  0.000000e+00
## grade6         2.39574527  0.06438877  37.20750336 5.160881e-303
## grade7         2.73054112  0.09186170  29.72447918 3.707233e-194
## emp_length_p -0.02209254  0.00275027  -8.03286275  9.522418e-16
## addr_stateAL   0.64719114  0.25709523   2.51732065  1.182512e-02
## addr_stateAR   0.61169892  0.26859207   2.27742731  2.276072e-02
## addr_stateAZ   0.65225638  0.25097080   2.59893337  9.351392e-03
## addr_stateCA   0.62645858  0.24352968   2.57241165  1.009927e-02
## addr_stateCO   0.19559979  0.25542421   0.76578406  4.438048e-01
## addr_stateCT   0.26759698  0.25967041   1.03052548  3.027634e-01
## addr_stateDC   0.29409502  0.33106510   0.88832987  3.743633e-01
## addr_stateDE   0.91846422  0.29521581   3.11116208  1.863526e-03
## addr_stateFL   0.65093020  0.24510127   2.65576021  7.912986e-03
## addr_stateGA   0.55004475  0.24861848   2.21240489  2.693870e-02
## addr_stateHI   0.70168219  0.27545761   2.54733277  1.085499e-02
## addr_stateIL   0.47213751  0.24768806   1.90617789  5.662713e-02
## addr_stateIN   0.62733644  0.25337087   2.47596121  1.328780e-02
## addr_stateKS   0.39681317  0.26787154   1.48135622  1.385117e-01
## addr_stateKY   0.60527141  0.26274651   2.30363253  2.124328e-02
## addr_stateLA   0.87816258  0.25596779   3.43075426  6.019056e-04
## addr_stateMA   0.70571832  0.25102534   2.81134296  4.933517e-03
## addr_stateMD   0.56652326  0.25082564   2.25863380  2.390617e-02
## addr_stateME -6.03449333 43.95462499  -0.13728916  8.908022e-01
## addr_stateMI   0.56836401  0.25052073   2.26873046  2.328472e-02
## addr_stateMN   0.69727831  0.25249690   2.76153210  5.753086e-03
## addr_stateMO   0.74038131  0.25373692   2.91790924  3.523869e-03
## addr_stateMS   0.74634028  0.27510085   2.71296974  6.668320e-03
## addr_stateMT   0.59918842  0.31016299   1.93185013  5.337801e-02
## addr_stateNC   0.63279067  0.24941238   2.53712618  1.117667e-02
## addr_stateNH   0.01511422  0.30495461   0.04956219  9.604713e-01
## addr_stateNJ   0.58117453  0.24763028   2.34694451  1.892807e-02
## addr_stateNM   0.92091993  0.27012642   3.40921828  6.514933e-04
## addr_stateNV   0.86038345  0.25472417   3.37770634  7.309309e-04
## addr_stateNY   0.70718540  0.24436557   2.89396492  3.804107e-03
## addr_stateOH   0.66208141  0.24776413   2.67222468  7.535018e-03
## addr_stateOK   0.57018951  0.26395859   2.16014761  3.076124e-02
## addr_stateOR   0.47260918  0.25984327   1.81882403  6.893828e-02
## addr_statePA   0.72359981  0.24745251   2.92419672  3.453464e-03
## addr_stateRI   0.58091326  0.28890566   2.01073689  4.435326e-02
## addr_stateSC   0.34438395  0.26188316   1.31502901  1.885002e-01
## addr_stateSD   0.48206928  0.32691599   1.47459683  1.403210e-01
## addr_stateTN   0.79969911  0.25315642   3.15891300  1.583588e-03
## addr_stateTX   0.57293600  0.24477585   2.34065573  1.924991e-02
## addr_stateUT   0.64015201  0.26829816   2.38597240  1.703403e-02
## addr_stateVA   0.63629442  0.24892368   2.55618273  1.058275e-02
## addr_stateVT   0.29364193  0.33929338   0.86545139  3.867911e-01
## addr_stateWA   0.50902538  0.25199027   2.02002002  4.338131e-02
## addr_stateWI   0.48016713  0.25935387   1.85139761  6.411237e-02
## addr_stateWV   0.38750054  0.28321469   1.36822186  1.712426e-01
## addr_stateWY   0.12178983  0.34690254   0.35107795  7.255299e-01
```

Interrupting scorecard

Using the significance level of 0.001 we can remove all coefficents with a p-value greater than this to see only the statistically significant coefficents of the model. We can the remove the intercept term and split these into positive and negative coefficents which shows their relationship with creditworthiness.

```
significant <- scorecard[scorecard[,4]<=0.001,]#remove if above significance level
significant <- significant[-1,] #remove intercept
significant_pos <- significant[significant[,1]>0,]#positive relationship
significant_neg <- significant[significant[,1]<0,]#negative relationship
significant_pos; significant_neg #print
```

```
##                Estimate Std. Error   z value      Pr(>|z|)
## loan_amnt    0.09680365 0.01740871  5.560646  2.687784e-08
## grade2       0.79513073 0.04953358 16.052356  5.503436e-58
## grade3       1.38150884 0.04785791 28.866884 3.110859e-183
## grade4       1.79186517 0.04933817 36.318031 8.402552e-289
## grade5       2.12105217 0.05307258 39.965122  0.000000e+00
## grade6       2.39574527 0.06438877 37.207503 5.160881e-303
## grade7       2.73054112 0.09186170 29.724479 3.707233e-194
## addr_stateLA 0.87816258 0.25596779  3.430754  6.019056e-04
## addr_stateNM 0.92091993 0.27012642  3.409218  6.514933e-04
## addr_stateNV 0.86038345 0.25472417  3.377706  7.309309e-04
```

```
##                 Estimate Std. Error   z value     Pr(>|z|)
## term60       -0.16789871 0.02566329 -6.542368 6.055213e-11
## emp_length_p -0.02209254 0.00275027 -8.032863 9.522418e-16
```

We can see the coefficents with a positive relation to defult and are significant are `loan_amnt`, `grade` for grades 2 to 7 not 1 and living in LA, NM or NV. The higher these are or if you belong to these catagories the less creditworthy you are. The significant coefficents with a negative relationship to defult are `term` being 60 and `emp_length_p`. So if you have a term length of 60 or have been employed for longer you are more creditworthy. This intuitively makes sense.

The most important predictor of creditworthiness is whether they are in grades 2 to 7 these are statistically a lot more significant and therefore important than the next most important predictor which is employement length then having a 60 month term. Following these is loan amount and then state from this we can conclude as a whole grade is the most important predictor, then employement length, then term, then loan amount and then state being the least important predictor of default.

ROC and AUC

We first create predictions on the test and training data using the logistic regression model we built earlier and a function to calculate the points of the ROC curve and plot them and to print the AUC.

```
ptrain <- predict(model, newdata=train)#Create predictions
ptest <- predict(model, newdata=test)#Create predictions

roc <- function(r,p,s){#make function to calculate roc and auc
  yav <- rep(tapply(r, p, mean), table(p))
  rocx <- cumsum(yav)
  rocy <- cumsum(1 - yav)
  area <- sum(yav * (rocy - 0.5 * (1 - yav)))
  x1 <- c(0, rocx)/sum(r)#calculate FPR
  y1 <- c(0, rocy)/sum(1 - r)#Calculate TPR
  auc <- area/(sum(r) * sum(1 - r))#Calculate AUC
  print(auc)#print auc
  plot(x1,y1,"l", main=s, xlab="FPR", ylab="TPR")#plot

}
```

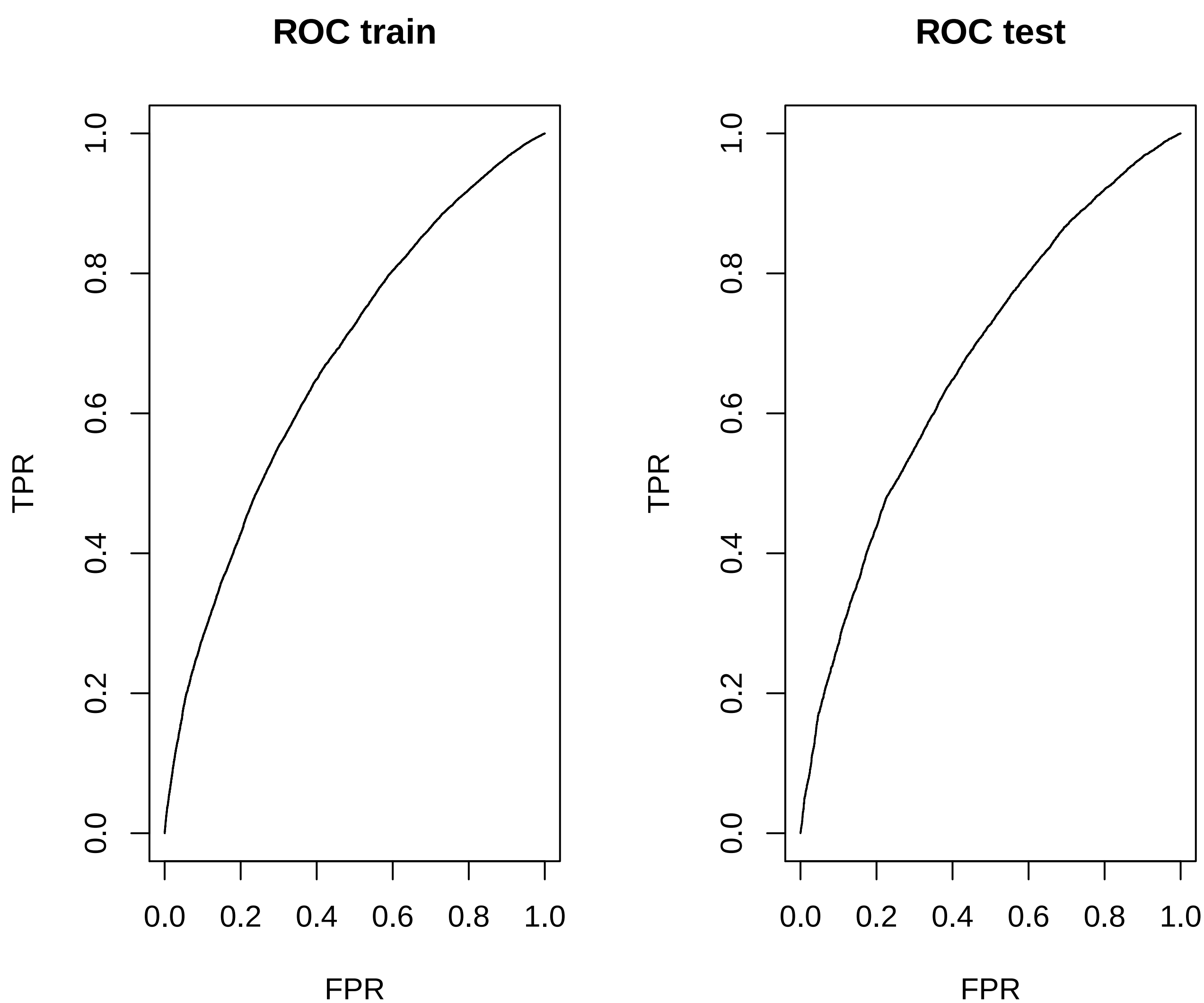
We then plot the two curves and show the area underneath them using the above function `roc`.

```r
par(mfrow=c(1,2))#make 1x2 space in plotting device
roc(train$def_flag,ptrain,"ROC train")#roc and auc training
```

```
## [1] 0.6729927
```

```r
roc(test$def_flag,ptest, "ROC test")#roc and auc testing
```

```
## [1] 0.6727498
```



The AUC for both the test and training data are similar although a bit lower for the test set however this may not be statistically significant. This would suggest the model has similar performance in any data set suggesting therefore that our model hasn't overfitted to the training dataset as the performance in the test set is very similar.

The AUC for both data sets is low suggesting the model isn't a good model for predicting default. It could be improved by either adding new predictors or using a penalised technique such as LASSO or Ridge as predctors like `emp_length_p` are heavily skewed.