Institute and Faculty of Actuaries

EXAMINERS' REPORT

CS2 - Risk Modelling and Survival Analysis

Core Principles

Paper B

April 2023

Introduction

The Examiners' Report is written by the Chief Examiner with the aim of helping candidates, both those who are sitting the examination for the first time and using past papers as a revision aid and also those who have previously failed the subject.

The Examiners are charged by Council with examining the published syllabus. The Examiners have access to the Core Reading, which is designed to interpret the syllabus, and will generally base questions around it but are not required to examine the content of Core Reading specifically or exclusively.

For numerical questions the Examiners' preferred approach to the solution is reproduced in this report; other valid approaches are given appropriate credit. For essay-style questions, particularly the open-ended questions in the later subjects, the report may contain more points than the Examiners will expect from a solution that scores full marks.

For some candidates, this may be their first attempt at answering an examination using open books and online. The Examiners expect all candidates to have a good level of knowledge and understanding of the topics and therefore candidates should not be overly dependent on open book materials. In our experience, candidates that spend too long researching answers in their materials will not be successful either because of time management issues or because they do not properly answer the questions.

Many candidates rely on past exam papers and examiner reports. Great caution must be exercised in doing so because each exam question is unique. As with all professional examinations, it is insufficient to repeat points of principle, formula or other text book works. The examinations are designed to test "higher order" thinking including candidates' ability to apply their knowledge to the facts presented in detail, synthesise and analyse their findings, and present conclusions or advice. Successful candidates concentrate on answering the questions asked rather than repeating their knowledge without application.

The report is written based on the legislative and regulatory context pertaining to the date that the examination was set. Candidates should take into account the possibility that circumstances may have changed if using these reports for revision.

Sarah Hutchinson
Chair of the Board of Examiners
July 2023

A. General comments on the *aims of this subject and how it is marked*

The aim of the Risk Modelling and Survival Analysis subject is to provide a grounding in mathematical and statistical modelling techniques that are of particular relevance to actuarial work, including stochastic processes and survival models.

Candidates are reminded of the need to include the R code, that they have used to generate their solutions, together with the main R output produced, in their answer script.

Where the R code was missing from a particular question part, no marks were awarded even if the output (e.g. a graph) was included. Partial credit was awarded in the cases where the R code was included but the R output was not.

The marking schedule below sets out potential R code solutions for each question. Other appropriate R code solutions gained full credit unless one specific approach had been explicitly requested in the question paper.

In cases where the same error was carried forward to later parts of the answer, candidates were given full credit for the later parts.

In higher order skills questions, where comments were required, well-reasoned comments that differed from those provided in the solutions also received credit as appropriate.

B. Comments on *candidate performance in this diet of the examination.*

It was noted by the examiners that performance on paper B in this diet was lower than was expected, with two of the three questions being found by candidates to be challenging and generally not well answered. Candidates should note that this examination is not a test of a set of R commands or packages, but rather an assessment of the candidates' ability to use R programming to solve problems in statistics and risk modelling. This requires a combination of three things:

- Understanding of all parts of the syllabus and core reading for this subject; please note that the syllabus is extensive, and the areas covered wide. Candidates are reminded to ensure that their preparation covers all areas of the syllabus including Stochastic Processes, Time Series and Machine Learning where it has been noted that questions are often not as well-answered as well as Survival Models and Loss Distributions.

- Competency in R programming including importing and manipulating data, performing calculations on vectors and matrices, producing plots and using a range of statistical functions.

- Practice in answering problem questions where the goal is to combine the first two elements above with analysis of a scenario or a data set to answer certain questions. Candidates are reminded that although this is an "open-book" examination, the time allowed, and this third element mean that candidates should prepare as if the examination was closed-book and using the exam time to concentrate on problem solving and the scenario presented.

C. Pass Mark

The Pass Mark for this exam was 51
1226 presented themselves and 376 passed.

Solutions for Subject CS2B - April 2023

Q1

(i)

```
transitions<-matrix(c(0.98, 0.019,0.001,0.2,0.75,
0.05,0,0,1),byrow=TRUE,nrow=3,dimnames=list
(c("healthy","sick",
"dead"), c("healthy","sick","dead")))
```
[1]

```
(transitions_markov <- new("markovchain", transition
Matrix=transitions))
```
[1]

A 3-dimensional discrete Markov Chain defined by the following states:
healthy, sick, dead

The transition matrix (by rows) is defined as follows:

healthy		sick	dead
healthy	0.98	0.019	0.001
sick	0.20	0.750	0.050
dead	0.00	0.000	1.000

(ii)

```
c(1,0,0)*transitions_markov^2
```
[1]

```
   healthy    sick    dead
[1,]  0.9642 0.03287 0.00293
```
[½]

The probability that the life will be sick at t=2 weeks is 3.287% [½]

(iii)

```
transitions_sick<-matrix(c(0.98, 0.019,0.001,0,1,
0,0,0,1),byrow=TRUE,nrow=3,
dimnames=list(c("healthy","sick","dead"),c("healthy",
"sick","dead")))
```
[1]

```
transitions_sickmarkov <- new("markovchain",
transitionMatrix=transitions_sick)
```
[½]

```
c(1,0,0)*transitions_sickmarkov^52
```
[½]

```
       healthy      sick       dead
[1,] 0.3497486 0.6177389 0.03251257
```
[½]

the probability that the life will be sick at some point in the next 52 weeks is
61.77% [½]

(iv)

```
0.98^52
```

The probability a healthy life will have remained healthy for the entire year is
34.97% [1]

(v)

It is reasonable to assume that the rate at which healthy lives become ill is proportional to the number of lives which are infectious.

or other similar reasonable comment about the nature of this transition [1]

(vi)
b reflects measures imposed for transmissivity e.g. masks, social distancing, lockdowns. A high b weak measures i.e. no masks, no social distancing [2]

(vii)
calculation of occupancy probabilities

```
#initial populations
s<-.01
h<-.99
r<-0
dead<-0                                   [½]
b<-.3                                     [½]
#par value

#daily rates
k<-.13                                    [½]
j<-.04                                    [½]

#just set up 4x4
days<-100                                 [½]
dt1=0.01                                  [½]
#in days
no_steps<-days/dt1                        [½]
#100 days
mat_ans<-matrix(0,no_steps,4)             [½]

for (qq in 1:no_steps) {                  [½]

  mu12=b*s
#mu12=b*s, or try constant rate
  mu13=0
  mu14=0                                  [½]

  mu21=0
  mu23=k #recovery rate
  mu24=j #death rate                      [½]

  mu31=0.0
  mu32=0.0
  mu34<-0                                 [½]

  mu43<-0
  mu41=0
  mu42=0                                  [½]

  mu11=-mu12-mu13-mu14
  mu22=-mu21-mu23-mu24
```



```
  mu33=-mu32-mu31-mu34
  mu44=-mu42-mu41-mu43                                          [½]

t_rates=c(mu11,mu12,mu13,mu14,mu21,mu22,mu23,mu24,mu31,
mu32,mu33,mu34,mu41,mu42,mu43,mu44)                             [½]
gen_matrix=matrix(t_rates,4,4,byrow=TRUE)                       [½]
Po=c(h,s,r,dead)                                                [½]

#starting proportions at each loop
P_dt =diag(4) + gen_matrix*dt1                                  [½]

#gets prob matrix each loop
occ_probs=Po%*%P_dt                                             [1]
#occupancy probs end of each period given starting states

  h<-occ_probs[1]
  s<-occ_probs[2]
  r<-occ_probs[3]
  dead<-occ_probs[4]                                            [½]
  mat_ans[qq,]<-occ_probs                                       [½]
#posts answers in matrix for plotting
```

(viii)

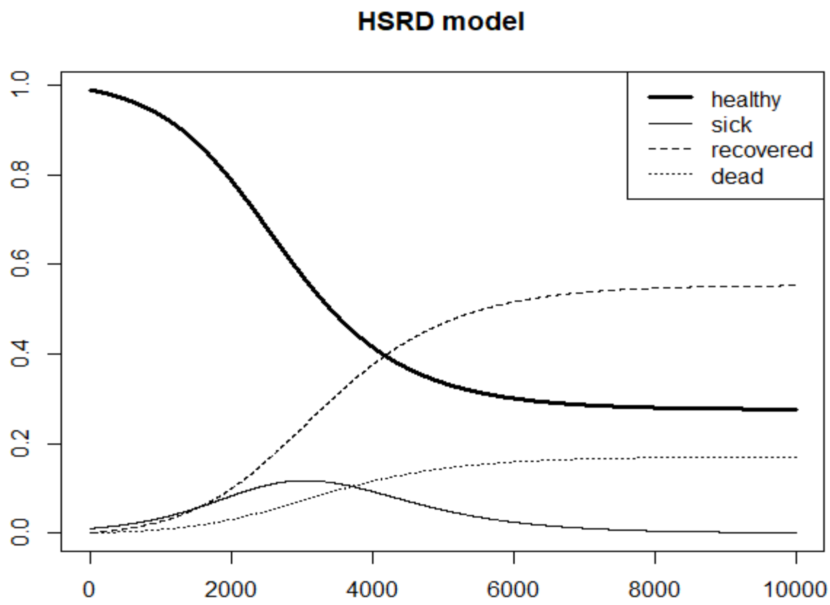
```
#plots
x = seq(from = 1/100, to = 100, by = 1/100)                     [1]

plot(x,mat_ans[,1],ylim=c(0,max(mat_ans)),col=1,lwd=3,
lty=1,xlab="",ylab="",type="l",main="HSRD model")               [1]
lines(x,mat_ans[,2],lwd=1,lty=1)                                [½]

#sick
lines(x,mat_ans[,3],lwd=1,lty=2)                                [½]

#recovered
lines(x,mat_ans[,4],lwd=1,lty=3)                                [½]

# dead
legend(x="topright",legend = c("healthy","sick",
"recovered","dead"),lwd = c(3,1,1,1),lty = c(1,1,2,3))          [½]
```



[1]

(ix)(a)

```
mat_ans[5/dt1,2]
[1] 0.01860525
```
[½] [½]

The probability that a life healthy at time 0 is sick after 5 days is 0.01860525 [½]

(b)

```
mat_ans[30/dt1,2]
[1] 0.1171159
```
[½] [½]

The probability that a life healthy at time 0 is sick after 30 days is 0.1171159 [½]

(x)

```
yyy<-mat_ans[,2]
sum(0.01*yyy*exp(-0.06*seq(1,10000)/36500))
[1] 4.227194
```
[1] [2] [½]

The required expected present value is £4.23 [½]

[Total 34]

This question on the application of Markov processes to survival analysis in R was not well answered. Candidates need to be ready to use R to solve problems and simply learning R commands will not be sufficient. Many candidates failed to see that a new matrix was required in part (iii).

Part (vii) was not well answered. Alternative methods for generating the mat_ans matrix are possible here and can obtain full marks. In general marks were be awarded in ½ mark increments for a wide variety of approaches in the following 5 stages to a solution for this part:
step 1 - parameterising the solution using figures given in the question (max 2 marks)

step 2 - setting the time sequence and initiating the loop (max 2 marks)
step 3 - calculating the mu values (max 3 marks)
step 4 - converting the mu's into probabilities (max 3 marks)
step 5 - outputting the matrix (1 mark)

Similarly for part (x) a wide variety of approaches to incorporating an interest rate element were given full credit.

Q2

(i)

```
Prem<-read.table("CS2B_A23_Qu_2_Data.csv", TRUE,
",")                                                    [1½]
Prem_charged=                                           [½]
2500*                                                   [1]
(Prem$Mort*Prem$p_dth*80000                             [1]
+Prem$Mort*(1-Prem$p_dth)*40000)                        [1]
Prem_charged                                            [½]
```

[1] 187500 260000 640000 080000 [½]

(ii)

```
set.seed(123)                                           [½]

Tab_R<-
data.frame(cbind(G1_R=rbinom(2500,1,Prem$Mort[1]),G2_
R=rbinom(2500,1,Prem$Mort[2]),G3_R=rbinom(2500,1,Prem$
Mort[3]),G4_R=rbinom(2500,1,Prem$Mort[4])))             [4½]
summary(Tab_R)                                          [½]
```

Or

```
set.seed(123)                                           [½]
G1_R=rbinom(2500,1,Prem$Mort[1])                        [1]
G2_R=rbinom(2500,1,Prem$Mort[2])                        [1]
G3_R=rbinom(2500,1,Prem$Mort[3])                        [1]
G4_R=rbinom(2500,1,Prem$Mort[4])                        [1]
Tab_R<- data.frame(cbind(G1_R,G2_R,G3_R,G4_R))          [½]
summary(Tab_R)                                          [½]
```

Or

```
set.seed(123)                                           [½]
B = matrix(nrow = 2500, ncol = 4)                       [½]
  for (i in 1:4) {                                      [½]
              q = Prem$Mort[i]                          [1½]
   B[ ,i] = rbinom(2500,1, q )                          [1½]
}
Tab_R=data.frame (G1_R=B[,1] , G2_R=B[,2], G3_R =B[,3],
G4_R=B[,4] )                                            [½]
 summary(Tab_R)                                         [½]
```


G1_R	G2_R	G3_R	G4_R	
Min. :0.0000	Min. :0.000	Min. :0.0000	Min. :0.0000	
1st Qu.:0.0000	1st Qu.:0.000	1st Qu.:0.0000	1st Qu.:0.0000	
Median :0.0000	Median :0.000	Median :0.0000	Median :0.0000	
Mean :0.0016	**Mean :0.002**	**Mean :0.0028**	**Mean :0.0068**	
3rd Qu.:0.0000	3rd Qu.:0.000	3rd Qu.:0.0000	3rd Qu.:0.0000	
Max. :1.0000	Max. :1.000	Max. :1.0000	Max. :1.0000	[½]

(The function "rbern" is not a part of the base R installation. However, if the candidate installs "Rlab" package and uses "rbern" function, full marks are to be awarded assuming no other errors)

install.packages("Rlab")
library("Rlab")
set.seed(123) [½]

Tab_R<-
data.frame(cbind(G1_R=rbern(2500,Prem$Mort[1]),G2_R=rbern(2500,Prem$Mort[2]),G3_R=rbern(2500,Prem$Mort[3]),G4_R=rbern(2500,Prem$Mort[4])) [4½]
summary(Tab_R) [½]

G1_	R G2_	R G3_	R G4_R	
Min. :0.0000	Min. :0.000	Min. :0.0000	Min. :0.0000	
1st Qu.:0.0000	1st Qu.:0.000	1st Qu.:0.0000	1st Qu.:0.0000	
Median :0.0000	Median :0.000	Median :0.0000	Median :0.0000	
Mean :0.0016	Mean :0.002	Mean :0.0028	Mean :0.0068	
3rd Qu.:0.0000	3rd Qu.:0.000	3rd Qu.:0.0000	3rd Qu.:0.0000	
Max. :1.0000	Max. :1.000	Max. :1.0000	Max. :1.0000	[½]

(iii)

```
set.seed(300)                                        [½]
U = matrix(runif(10000), ncol = 4)                   [2]
Tab_U=data.frame (G1_U=U[,1] , G2_U=U[,2], G3_U =U[,3],
G4_U=U[,4] )                                         [½]
head(Tab_U)                                          [½]
```

Or

```
set.seed(300)                                        [½]
 U = matrix(nrow = 2500, ncol = 4)                   [½]
  for (j in 1:4) {                                   [½]
     U[ ,j] = runif(2500)                            [1]
            }
Tab_U=data.frame (G1_U=U[,1] , G2_U=U[,2], G3_U =U[,3],
G4_U=U[,4] )                                         [½]
head(Tab_U)                                          [½]
```

	G1_U	G2_U	G3_U	G4_U
1	0.9152467	0.9633111	0.4436952	0.4876556
2	0.7633293	0.5037355	0.3548937	0.9567698
3	0.8056856	0.8878896	0.5736878	0.3526416

4	0.7337780	0.3721702	0.5864507	0.8759781	
5	0.6820679	0.6179086	0.8117664	0.5643849	
6	0.0120303	0.3733971	0.4308092	0.2999071	[½]

(iv)

```
M = matrix(nrow = 2500, ncol = 4)
dth1 =80000                                     [½]
dth2 =40000                                     [½]
  for (i in 1:4) {                              [1]
   M[,i] =                                      [½]
Tab_R[,i]  *                                    [1]
ifelse(                                         [1]
Tab_U[,i]<= Prem$p_dth[i],                      [1½]
dth1,                                           [½]
dth2)                                           [½]

     }
Tab_V=data.frame (G1_V=M[,1] , G2_V=M[,2], G3_V =M[,3],
G4_V=M[,4] )                                    [½]
summary(Tab_V)                                  [½]
```

```
                 G1_V             G2_V             G3_V  G4_V
Min.   :   0     Min.   :   0     Min.   :   0     Min.   :   0
1st Qu.:   0     1st Qu.:   0     1st Qu.:   0     1st Qu.:   0
Median :   0     Median :   0     Median :   0     Median :   0
Mean   :  96     Mean   :  96     Mean   : 192     Mean   : 464
3rd Qu.:   0     3rd Qu.:   0     3rd Qu.:   0     3rd Qu.:   0Max.
:80000Max.   :80000          Max.   :80000         Max.   :80000   [½]
```

(v)

```
Aggregate<- c(sum(Tab_V$G1_V) , sum(Tab_V$G2_V),
sum(Tab_V$G3_V), sum(Tab_V$G4_V) )              [1]
Aggregate                                       [1]
1,] 240000 240000 480000 1160000                [1]
```

(vi)

```
Prem_charged-Aggregate                          [½]
[1]   -52500  20000 160000-80000                [½]
sum(Prem_charged-Aggregate)                     [½]
[1] 47500                                       [½]
```

(vii)
The employee's suggestion is not valid [1]
as the aggregate claim amount in part (v) under each group represents one sample [1½]
The above process would have to be performed many times, and averages taken, to obtain reasonable premium rates [1½]

[Total 33]

This question was well answered, and the average mark was considerably higher than that for the other two questions on the paper.

In part (i) some candidates forgot to multiply by 2500 policies to produce total premiums and instead calculated premiums per policy.

In part (ii) use of `rbern` rather than `rbinom` gained full marks if applied correctly. Candidates are reminded to include the code for setting the seed value in these simulations questions. The examiners understand that where a candidate has the seed but needs to run their code more than once to get a correct answer then the numerical values produced will be different to those above. Candidates are not penalised for this.

In part (vii) marks were awarded for a range of sensible arguments in support of the correct conclusion.

Q3

```
install.packages("rpart")
install.packages("rpart.plot")
library(rpart)
library(rpart.plot)
```

(i)

```
set.seed(123)                                                    [½]
A = matrix(nrow = 100000, ncol = 5)                              [½]
  A[,1] = runif(n = 100000)                                      [1]
  A[,2] = runif(n = 100000)                                      [½]
  A[,3] = rlnorm(n = 100000, meanlog = 7.5, sdlog = 0.5)         [1]
  A[,4] = rlnorm(n = 100000, meanlog = 7, sdlog = 0.5)           [1]
  A[,5] = runif(n = 100000)                                      [½]
  head(A)                                                        [½]

          [,1]       [,2]     [,3]      [,4]      [,5]
[1,] 0.2875775 0.60240988 2064.197 1267.1684 0.4708851
[2,] 0.7883051 0.02285169 4515.963  728.4591 0.8977670
[3,] 0.4089769 0.82056246 1755.152  512.0199 0.7615472
[4,] 0.8830174 0.03656945 1760.574  884.9737 0.6478515
[5,] 0.9404673 0.23504938 2250.600  703.6842 0.5329256
[6,] 0.0455565 0.87053886 3528.840 1001.6978 0.8526255
                                                                 [½]
```

(ii)

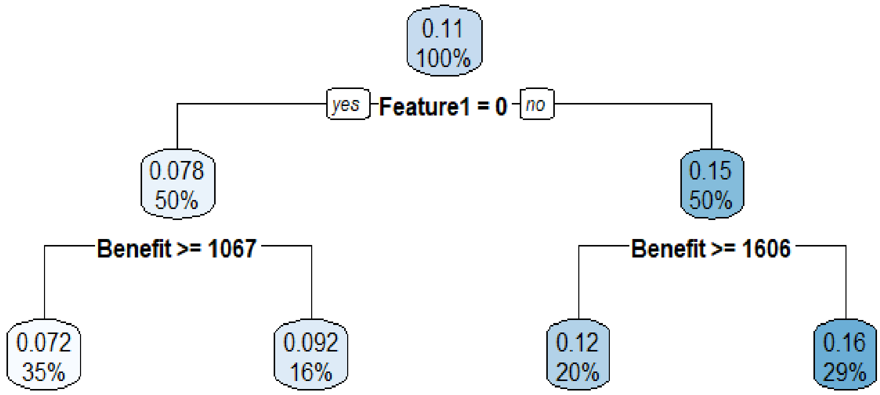
```
Feature1 = ifelse(A[,1] < 0.5, 0, 1)                             [2]
Feature2 = ifelse(A[,2] < 0.5, 0, 1)                             [½]
Benefit = ifelse(Feature2 == 0, A[,3], A[,4])                    [1½]
Outcome0prob = ifelse(Feature1 == 0 & Feature2 == 0,
0.95, ifelse(Feature1 == 1 & Feature2 == 1, 0.8, 0.9))           [2]
```

```
Outcome = ifelse(A[,5] < Outcome0prob, 0, 1)                    [1]
B = cbind(Feature1, Feature2, Benefit, Outcome)                 [1]
head(B)                                                         [½]
     Feature1 Feature2   Benefit Outcome
[1,]        0        1 1267.1684       0
[2,]        1        0 4515.9630       0
[3,]        0        1  512.0199       0
[4,]        1        0 1760.5743       0
[5,]        1        0 2250.5999       0
[6,]        0        1 1001.6978       0
```

[½]

(iii)

```
tree = rpart(formula = Outcome ~ Feature1 + Benefit,
data = data.frame(B), control = rpart.control(cp = 2e-4,
maxdepth = 2, minbucket = 5000))
rpart.plot(tree)                                                [½]
```



[½]

(iv)
The tree is showing that customers with higher benefit amounts should be eligible for the simplified underwriting process in preference to those with lower benefit amounts [1]
This has arisen because customers with low benefit amounts are more likely to have *Feature2* = 1 [1]
and hence to have a higher probability of an adverse outcome under the existing underwriting process [1]
For benefit amount to affect underwriting decisions in this way is inconsistent with the spirit of the regulation prohibiting the use of *Feature2* as an underwriting criterion [1]
The tree is therefore unsuitable for practical use [1]

(v)
In the data set *B*, observations appear with a likelihood based on the distribution of *Benefit* conditional on the actual values of *Feature2* [1]

With the specified weights, observations effectively appear with a likelihood based on the unconditional distribution of *Benefit*, without regard to the values of *Feature2* [1]

(1 mark for any remark that relates use of weights to the Features)

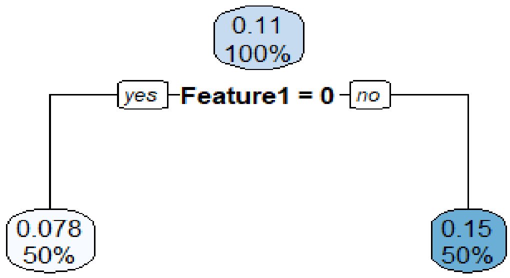
(vi)

```
Weight = vector(length = 100000)                                    [½]
for(i in 1:100000) {                                                [½]
   w0i = dlnorm(Benefit[i], meanlog = 7.5, sdlog = 0.5)             [1]
   w1i = dlnorm(Benefit[i], meanlog = 7, sdlog = 0.5)               [½]
   wi = ifelse(Feature2[i] == 0, w0i, w1i)                          [1]
   Weight[i] = (w0i + w1i) / 2 / wi                                 [½]
}

head(Weight)                                                        [½]
[1] 1.117407 5.642813 4.281520 1.281643 1.777311 1.488022           [½]
```

(vii)

```
tree2 = rpart(formula = Outcome ~ Feature1 + Benefit,
data = data.frame(B), weights = Weight,
control = rpart.control(cp = 2e-4, maxdepth = 2,
minbucket = 5000))
rpart.plot(tree2)                                                   [½]
```



[½]

(viii)

The weighting has successfully eliminated the dependency on *Benefit* arising from the confounding feature *Feature2* [1]

In practice, there will be many more features in the data and trees as simple as *tree* and *tree2* are unlikely to be used [1]

In practice, there will be greater complexity in assessing whether the effect of a given feature is due to a confounding feature such as *Feature2* which is not allowed to be used as a criterion [1]

The application of judgement is therefore likely to be required, rather than directly using the output of the *rpart* algorithm [1]

For the same probability of an adverse outcome of full medical underwriting, the simplified process is in practice less likely to be appropriate for large benefit amounts, because of the greater financial impact of an inappropriate decision [1]

One method of allowing for this would be to use *Benefit* * *Outcome* in place of *Outcome* in constructing *tree2* [1]

In practice, the large number of available features is likely to mean that a more sophisticated algorithm, such as the random forest or gradient boosting algorithm, is appropriate [1]

General comments about the application of decision trees *[1 per point, maximum 2]*

Discussion of alternatives to log Normal weights [1]

[Marks available 10, maximum 4]

[Total 33]

This question was not well answered. Candidates are reminded of the importance of including the syllabus section on Machine Learning techniques in their preparation. Questions in this area, which are most likely to be examined in the R programming paper given the nature of Machine Learning, continue to be poorly answered. This question is also a reminder to candidates that the examination tests understanding of the subject areas not necessarily knowledge of particular R packages. Candidates who understood the application of decision trees and had basic understanding of R could perform well on this question.

In part (iv) marks were awarded for a wide variety of sensible comments about decision trees and what the plots generated by the code given in the question were suggesting.

Similarly in part (v) credit was given for comments that related the calculation performed back to the scenario in the question.

[Paper Total 100]

END OF EXAMINERS' REPORT

Institute
and Faculty
of Actuaries

Beijing

14F China World Office 1 · 1 Jianwai Avenue · Beijing · China 100004
Tel: +86 (10) 6535 0248

Edinburgh

Level 2 · Exchange Crescent · 7 Conference Square · Edinburgh · EH3 8RA
Tel: +44 (0) 131 240 1300

Hong Kong

1803 Tower One · Lippo Centre · 89 Queensway · Hong Kong
Tel: +852 2147 9418

London (registered office)

7th Floor · Holborn Gate · 326-330 High Holborn · London · WC1V 7PP
Tel: +44 (0) 20 7632 2100

Oxford

1st Floor · Belsyre Court · 57 Woodstock Road · Oxford · OX2 6HJ
Tel: +44 (0) 1865 268 200

Singapore

5 Shenton Way · UIC Building · #10-01 · Singapore 068808
Tel: +65 8778 1784

www.actuaries.org.uk