INSTITUTE OF ACTUARIES OF INDIA

Subject CS2 - Risk Modelling and Survival Analysis (Paper B)

May 2024 Examination

INDICATIVE SOLUTION

Introduction

The indicative solution has been written by the Examiners with the aim of helping candidates. The solutions given are only indicative. It is realized that there could be other points as valid answers and examiner have given credit for any alternative approach or interpretation which they consider to be reasonable.

Solution 1:

(i)

- Informative censoring is likely to be present.
- The deaths for an unknown reason may or may not have been from blood clots.
- Even if the deaths for an unknown reason were not from blood clots, they are still likely to constitute informative censoring.
- This is because, had these lives not died, they were likely to have been in poorer health, and hence more likely to suffer from blood clots, than those remaining.

(3)

(ii)

```
install.packages("survival")
library(survival)
datamain<-read.csv(file.choose())
>ST<-ifelse(datamain$Status==2,1,0)
>datanew<- cbind(datamain, ST)
>tail(datanew,20)
```

	Life	Drug	Age	co_morbidity	Status	Time	ST
2481	2481	1	1	0	0	30	0
2482	2482	1	1	0	0	30	0
2483	2483	1	1	0	0	30	0
2484	2484	1	1	0	0	30	0
2485	2485	1	1	0	2	12	1
2486	2486	1	1	0	0	30	0
2487	2487	1	1	1	0	30	0
2488	2488	1	1	1	0	30	0
2489	2489	1	1	0	0	30	0
2490	2490	1	1	0	0	30	0
2491	2491	1	1	0	0	30	0
2492	2492	1	1	1	0	30	0
2493	2493	1	1	0	0	30	0
2494	2494	1	1	0	0	30	0
2495	2495	1	1	0	0	30	0
2496	2496	1	1	1	2	8	1
2497	2497	1	1	1	0	30	0
2498	2498	1	1	0	0	30	0
2499	2499	1	1	0	0	30	0
2500	2500	1	1	0	0	30	0

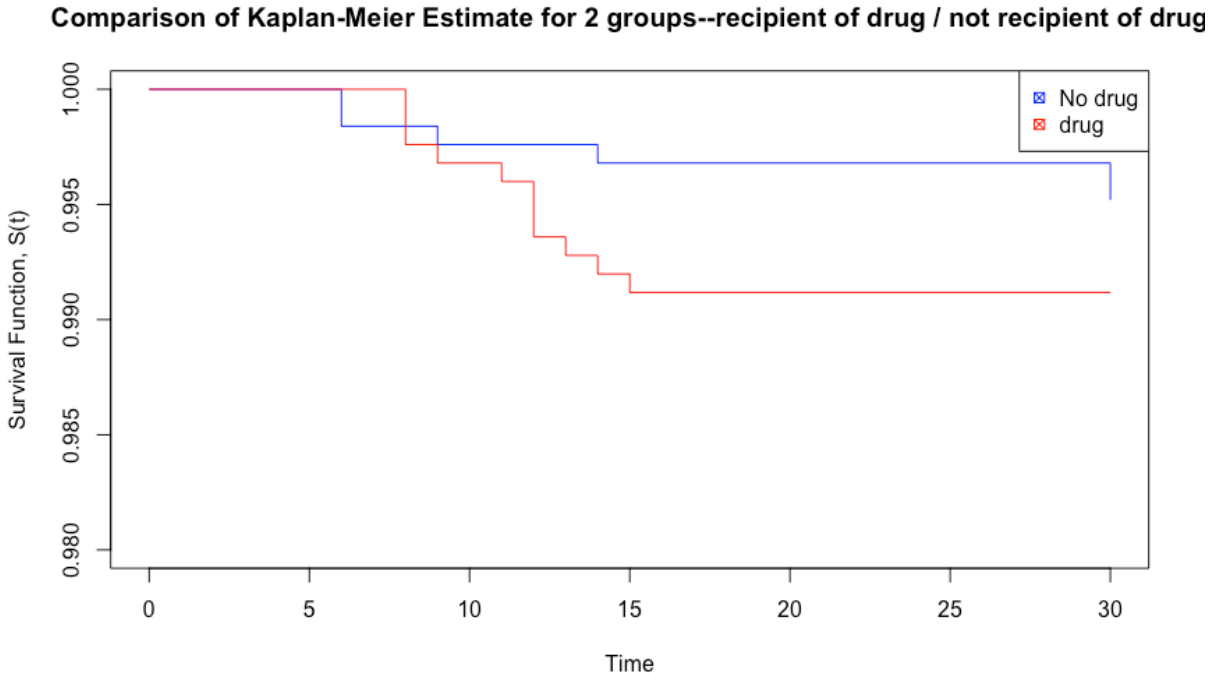
(1 mark for loading the library, 3 marks for correct code and 2 mark for correct output.)

(6)

(iii)

```
 S = survfit(
 +           Surv(datanew$Time, datanew$ST)
 +           ~datanew$Drug)
 > plot(
 +      S,
 +      xlab = "Time",
 +      ylab = "Survival Function, S(t)",
 +      ylim=c(.98,1),
 +      col = c("blue", "red") ,
```

```
+       main = "Comparison of Kaplan-Meier Estimate for 2 groups--recipient of drug / not recip
ient of drug")
> legend("topright", legend = c("No drug", "drug") ,
+       col = c("blue", "red")
+       , pch =7)
```



(6)

(iv)

- Individuals who have not been administered drug have a lower possibility of blood clots within a 30 day period than Individuals who received drug.
- Any effect of drug occurs, if at all, occurs within the first 15 days.
- The survival curves cross each other early in the curve.
- In general, lines crossing each other may mean violation of proportionality in hazard rate. Here, it may be insignificant due to small sample size and/or small number of events occurring.
- However, analysis does not consider possibility of other factors affecting the results.

(1 mark for each comment, max 4 marks)

(v)

H0: co-morbidity has no significant impact on blood clots along with vaccine indicator and age

H1: co-morbidity has significant impact on blood clots along with vaccine indicator and age

```
cox_1<-
+ coxph(
+ Surv(datanew$Time,datanew$ST)
+       ~ datanew$Drug*datanew$Age,
+       ties = "breslow")
cox_2<-coxph(Surv(datanew$Time,
+ datanew$ST)~datanew$Drug*datanew$Age*datanew$co_morbidity, ties = "breslow")
L1<-cox_1$loglik[2]
L2<-cox_2$loglik[2]
2 *( L2- L1)
[1] 15.50062
```

```
qchisq(0.95, 4)
[1] 9.487729
```

Conclusion: We reject Ho as the effect is statistically significant and conclude that co-morbidity along with drug and age has an impact on blood clots.

(7)
[26]

Solution 2:

(i)

```
set.seed(1234)
>
> V = matrix(runif(60000), ncol = 3)
> head(V)
          [,1]      [,2]       [,3]
[1,] 0.1137034 0.0346164 0.04571879
[2,] 0.6222994 0.8765384 0.03803062
[3,] 0.6092747 0.7347251 0.16973801
[4,] 0.6233794 0.6218362 0.15442048
[5,] 0.8609154 0.6978030 0.25847399
[6,] 0.6403106 0.5783307 0.86357008
```

(3)

(ii)

```
a = 0
b = 0.5
c = 0.5
n = 20000
w1 = vector(length = n)
w2 = vector(length = n)
w = vector(length = n)
defaultprob = vector(length = n)
default = vector(length = n)
for (i in 1:n) {
+ w1[i] = -1 *(V[i, 1] <= 1/3) + 1 * (V[i, 1] > 2/3)
+ w2[i] = 2 * (V[i, 2] - 0.5)
+ w[i] = a + b * w1[i] + c * w2[i]
+ defaultprob[i] = exp(w[i]) /(1 + exp(w[i]))
+     default[i] = 1 * (V[i, 3] <= defaultprob[i])
+ }
sample=data.frame("w1" = w1, "w2" = w2, "default" = default)
head(sample)
```

```
  w1         w2 default
1 -1 -0.9307672       1
2  0  0.7530768       1
3  0  0.4694501       1
4  0  0.2436723       1
5  1  0.3956060       1
6  0  0.1566614       0
```

(10)

(iii)

```
mean(defaultprob)
```

[1] 0.5006632

The expected probability of customer default is 0.5006632

The answer is not likely to be realistic as the mean default probability of their potential customers would be desired to be significantly less than 0.5. A negative value of *a* may reduce the expected probability of default.

(4)

(iv)

```
set.seed(1234)
V = matrix(runif(60000), ncol = 3)
a =- 0.6
> b = 0.5
> c = 0.5
> n = 20000
> w1 = vector(length = n)
> w2 = vector(length = n)
> w = vector(length = n)
> defaultprob = vector(length = n)
> default = vector(length = n)
>
>
> w1 = -1 *(V[,1] <= 1/3) + 1 * (V[,1] > 2/3)
> w2 = 2 * (V[,2] - 0.5)
> w = a + b * w1 + c * w2
> defaultprob = exp(w) /(1 + exp(w))
> default = 1 * (V[,3] <= defaultprob)
mean(defaultprob)
[1] 0.3627064
```

The mean default probability has come down significantly as expected

(4)

(v)

```
set.seed(1234)
 V = matrix(runif(60000), ncol = 3)
 a = 0
 b = 0.5
 c = 0.5
 n = 20000
  w1 = vector(length = n)
 w2 = vector(length = n)
  w = vector(length = n)
  defaultprob = vector(length = n)
  default = vector(length = n)

 w1 = -1 *(V[,1] <= 1/3) + 1 * (V[,1] > 2/3)
 w2 = 2 * (V[,2] - 0.5)
 w = a + b * w1 + c * w2
defaultprob = 0.5*exp(w) /(1 +0.5* exp(w))

> default = 1 * (V[,3] <= defaultprob)
> mean(defaultprob)
[1] 0.3425714
```

The result has improved as the default probability has become reasonable.

(4)

(vi) actual = read.csv(file.choose())
>head(actual, 10)

	w1	w2	default
1	-1	0.4842612	0
2	1	0.8991797	0
3	0	-0.2146289	0
4	1	-0.7641544	0
5	-1	-0.5532892	0
6	1	-0.9808636	1
7	-1	-0.8397735	0
8	-1	0.5445384	0
9	0	0.9868044	0
10	1	0.3109878	1

(2)

(vii)

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

a = 0
b = 0.5
c = 0.5
n = 20000
w1 = vector(length = n)
w2 = vector(length = n)
w = vector(length = n)
defaultprob = vector(length = n)
default = vector(length = n)
for (i in 1:n) {
  w1[i] = -1 *(V[i, 1] <= 1/3) + 1 * (V[i, 1] > 2/3)
  w2[i] = 2 * (V[i, 2] - 0.5)
  w[i] = a + b * w1[i] + c * w2[i]
  defaultprob[i] = exp(w[i]) /(1 + exp(w[i]))
  default[i] = 1 * (V[i, 3] <= defaultprob[i])
}
sample=data.frame("w1" = w1, "w2" = w2, "default" = default)

tree = rpart(default ~ ., data = sample, method = "class")

predict_defaults = predict(tree, actual, type = 'class')

> head(predict_defaults, 20)
 1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20
 0  1  0  1  0  1  0  0  1  1  1  0  1  1  0  1  0  0  1  0
Levels: 0 1
```

(4)

(viii)

```
Confusion_matrix = table(actual$default,predict_defaults);
>Confusion_matrix
predict_defaults
```

```
   0  1
0 63 37
1  0 25
```

(2)

(ix)

```
precision = Confusion_matrix[2,2] / sum(Confusion_matrix[,2])
>precision
[1] 0.4032258
recall = Confusion_matrix[2,2] / sum(Confusion_matrix[2,]);
>recall
[1] 1
```

The precision percentage is 40.32% and the recall percentage is 100%

(4)

(x)

- The recall percentage is the percentage of defaults that the model managed to identify. Here the model has performed well and has identified all 25 actual defaults. However, the model is currently not very precise. The precision percentage is the percentage of predicted defaults that are in fact actual defaults. The model has predicted far more defaults than was actually the case.

- Hence, if this model had been used to approve the loans of these 125 customers, 30% of them would have not been approved for a loan even though they did not actually default. The model is therefore not commercially optimised.

- This is in line with our conclusions from part (iii) i.e. that the probability of default in the specimen data used to train the model was unrealistically too high.

- The Analyst could refine the parameters (*a*, *b*, *c*) used to construct the specimen data and re-train the model to improve it. Better still, further data could be gathered from the public domain, if available, and used to train the model to improve it.

- Additionally, other decision tree models (e.g. bagged decision trees, random forests, boosted decision trees) may be investigated to see if a better fit can be obtained.

- Alternatively, other classification machine learning models (e.g. naïve Bayes classification) should be investigated to see if a better fit can be obtained.

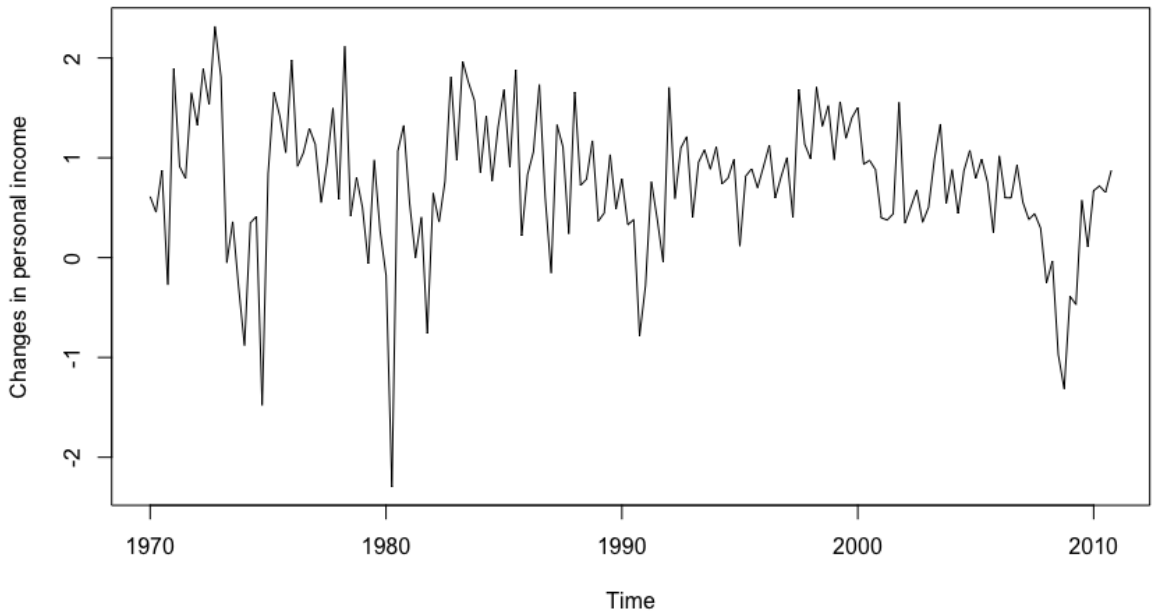
- Additionally, the Analyst could change the approach from classification of loan default/not default to a probability of loan default approach and use some form of regression machine learning algorithm to predict the probabilities.

(6)
[43]

Solution 3:

(i)

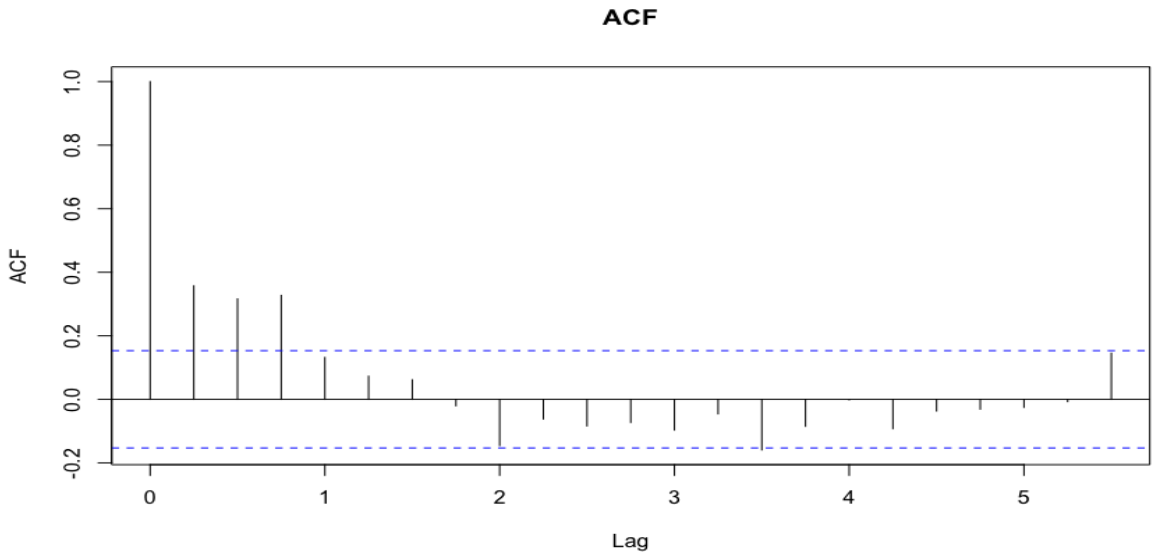
```
install.packages("fpp")
library(fpp)
consumption = usconsumption[ ,1]
plot(consumption, ylab = "Changes in personal income")
```



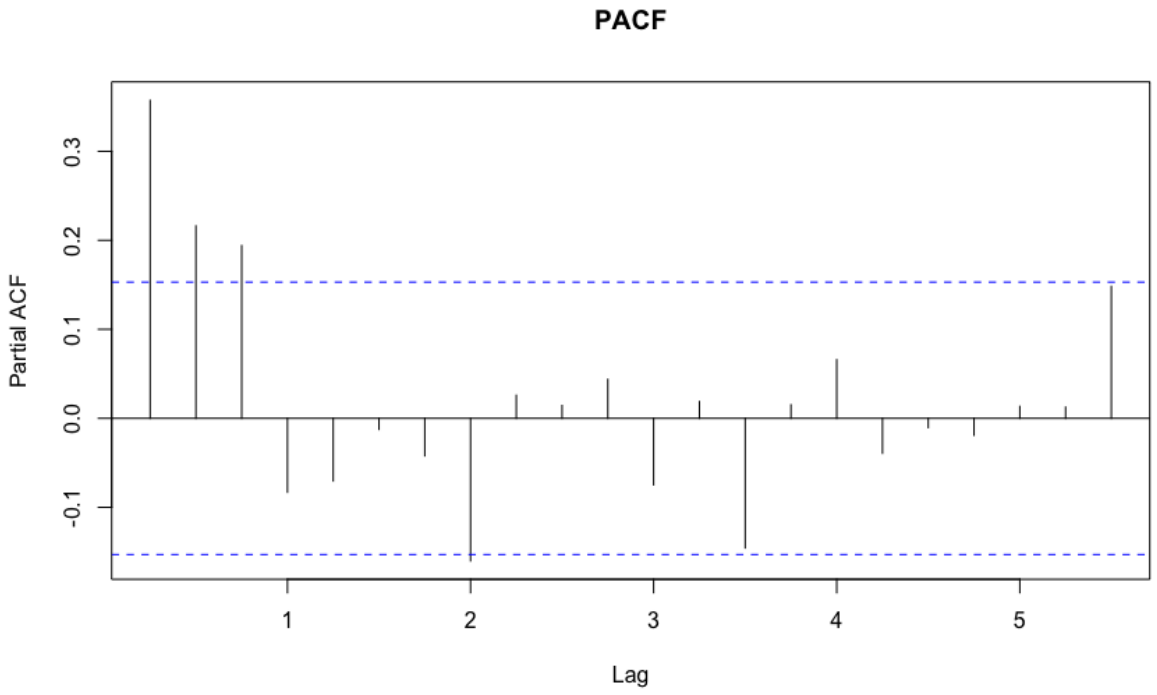
(3)

(ii)

> acf(consumption, main = "ACF")



pacf(consumption, main = "PACF")



(4)

(iii)

```
model1 <- arima(x = consumption, order = c(3, 0, 0))
model1

Call:
arima(x = consumption, order = c(3, 0, 0))

Coefficients:
         ar1     ar2     ar3  intercept
      0.2366  0.1603  0.1909     0.7533
s.e.  0.0763  0.0774  0.0759     0.1153

sigma^2 estimated as 0.3825:  log likelihood = -154.08,  aic = 318.16
```

In this case, the equation of the model is:

Xt=0.7533 + 0.2366 X_{t-1} + 0.1603 X_{t-2} + 0.1909 X_{t-3} + εt

(4)

(iv)

```
income<-usconsumption[,2]

# forecast from the AR(3)
forecast1 <- fitted(model1)
# fit and forecast linear regression
> model3 <- lm(consumption ~ income, data = usconsumption)
> forecast3 <- fitted(model3)
# Calculate RMSE
> n <- length(forecast1)
> rmse1 <- sqrt( sum((forecast1 - consumption)^2)/n )
> rmse1
[1] 0.6185039
rmse3 <- sqrt( sum((forecast3 - consumption)^2)/n )
```

```
> rmse3
[1] 0.6236113
```

(8)

(v)

(a)

It seems reasonable to consider a regression model with ARIMA which essentially blends the two models proposed in (iii) and (iv).

(b)

```
model4 <- arima(x=consumption, order = c(3, 0, 0), xreg = income)
```

(c)

```
> AIC(model4)
[1] 300.58
```

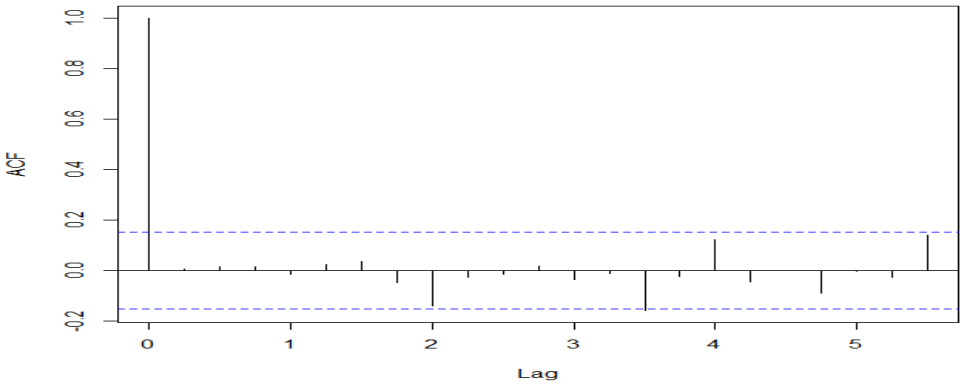
Hence model4 is better in terms of AIC

```
> forecast4 <- fitted(model4)
> rmse4 <- sqrt( sum((forecast4 - consumption)^2)/n )
 rmse4
[1] 0.582791
```

Hence, model4 fits the data better as measured by the RMSE

The residuals from model4 look better as shown below.

```
residuals4 <- residuals(model4)
acf(residuals4,main="ACF of residuals of suggested model")
```



(12)

[31]

****