

INSTITUTE OF ACTUARIES OF INDIA

EXAMINATIONS

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

Time allowed: 1 Hour 45 Minutes

Total Marks: 100

Q. 1)

```
install.packages("dplyr", type = "binary")
install.packages("randomForest", type = "binary")
install.packages("caret", type = "binary")
install.packages("rpart", type = "binary")
install.packages("rpart.plot", type = "binary")
```

```
library(dplyr)
library(randomForest)
library(caret)
library(rpart)
library(rpart.plot)
```

i)

```
heart_disease <- read.csv("<path>/heart_disease.csv")

heart_disease$sex <- as.factor(heart_disease$sex)
heart_disease$cp <- as.factor(heart_disease$cp)
heart_disease$fbs <- as.factor(heart_disease$fbs)
heart_disease$restecg <- as.factor(heart_disease$restecg)
heart_disease$exang <- as.factor(heart_disease$exang)
heart_disease$slope <- as.factor(heart_disease$slope)
heart_disease$ca <- as.factor(heart_disease$ca)
heart_disease$thal <- as.factor(heart_disease$thal)
heart_disease$target <- as.factor(heart_disease$target)
```

Alternate Solution

```
heart_disease <- heart_disease %>% mutate(across(c(sex, cp, fbs, restecg, exang, slope, ca,
thal, target), factor))
```

(3)

ii)

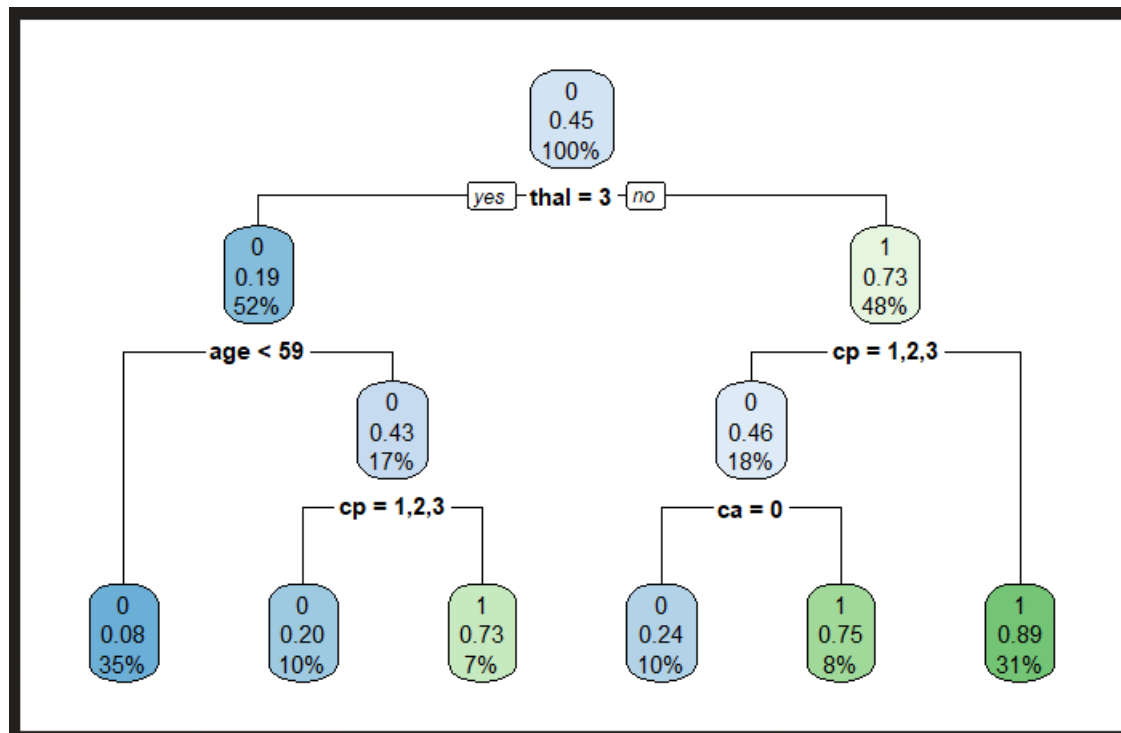
```
train_index <- c(1:(nrow(heart_disease)*0.7))
train_data <- heart_disease[train_index, ]
test_data <- heart_disease[-train_index, ]
```

(2)

iii)

```
set.seed(123)
tree_model <- rpart(target ~ age + sex + cp + trestbps + chol + fbs + restecg + thalach + exang
+ oldpeak + slope + ca + thal,data = train_data, method = "class")

rpart.plot(tree_model)
```



Inferences

- Patients have been split basis the value of thalassemia. If the value is less than 3, then they go to left node else to the right node. Thus, 45% of the data has moved to right node and remaining to the left node.
- On the left node, a further split has been done basis the age of the patient. If the age is less than 59, then it has branched leftward. This represents 35% of the total data. Thus, the probability that a heart disease is diagnosed when thalassemia is 3 and age of the patient is less than 59 is 0.08. However, with thalassemia being 3, age being greater than 59 and chest pain with 4, the probability of diagnosis the heart diseases is 0.73
- The right node has been split basis the value of the chest pain. If the value is less than or equal to 3, it has been further split basis the number of major vessels.
- Thus, if the thalassemia is not normal but, chest pain being less than equal to 3 and number of colored vessels being 0, the probability of getting heart diseases diagnosed is 0.24. The probability increases to 0.75 if number of colored vessels is greater than 0.
- However, if thalassemia is not normal and chest pain is 4, the probability of getting heart diseases diagnosed increases to 0.89. 31% of total population falls here.

(3)

iv)

```
pred_tree <- predict(tree_model, test_data, type = "class")
```

```
confusionMatrix(pred_tree, test_data$target)
```

Confusion Matrix and Statistics

```

Reference
Prediction 0 1
0 39 14
1 7 30

```

```

Accuracy : 0.7667
95% CI : (0.6657, 0.8494)
No Information Rate : 0.5111

```

P-Value [Acc > NIR] : 5.619e-07

Kappa : 0.5315

Mcnemar's Test P-Value : 0.1904

Sensitivity : 0.8478

Specificity : 0.6818

Pos Pred Value : 0.7358

Neg Pred Value : 0.8108

Prevalence : 0.5111

Detection Rate : 0.4333

Detection Prevalence : 0.5889

Balanced Accuracy : 0.7648

'Positive' Class : 0

```
confusion_matrix_tree <- confusionMatrix(pred_tree, test_data$target)
```

```
confusion_matrix_tree$overall['Accuracy']
```

```
Accuracy  
0.7666667
```

```
confusion_matrix_tree$byClass['Precision']
```

```
Precision  
0.7358491
```

```
confusion_matrix_tree$byClass['Recall']
```

```
Recall  
0.8478261
```

(4)

v)

```
set.seed(123)
```

```
rf_model_1 <- randomForest(target ~ age + sex + cp + trestbps + chol + fbs + restecg + thalach + exang  
+ oldpeak + slope + ca + thal,data = train_data, ntree = 100, mtry = 13)
```

```
pred_rf_1 <- predict(rf_model_1, test_data)
```

```
confusionMatrix(pred_rf_1, test_data$target)
```

Confusion Matrix and Statistics

Reference

Prediction 0 1

0 42 14

1 4 30

Accuracy : 0.8

95% CI : (0.7025, 0.8769)

No Information Rate : 0.5111

P-Value [Acc > NIR] : 1.329e-08

Kappa : 0.5978

Mcnemar's Test P-Value : 0.03389

Sensitivity : 0.9130
 Specificity : 0.6818
 Pos Pred Value : 0.7500
 Neg Pred Value : 0.8824
 Prevalence : 0.5111
 Detection Rate : 0.4667
 Detection Prevalence : 0.6222
 Balanced Accuracy : 0.7974

'Positive' Class : 0

(4)

vi)

The default value of mtry is $d/3$ or $\sqrt{d} \sim 4$ (rounded off)

```
set.seed(123)
```

```
rf_model_2 <- randomForest(target ~ age + sex + cp + trestbps + chol + fbs + restecg + thalach + exang + oldpeak + slope + ca + thal, data = train_data, ntree = 100)
```

```
pred_rf_2 <- predict(rf_model_2, test_data)
confusionMatrix(pred_rf_2, test_data$target)
```

Confusion Matrix and Statistics

		Reference	
Prediction		0	1
0	41	12	
1	5	32	

Accuracy : 0.8111
 95% CI : (0.7149, 0.8859)
 No Information Rate : 0.5111
 P-Value [Acc > NIR] : 3.351e-09

Kappa : 0.6207

Mcnemar's Test P-Value : 0.1456

Sensitivity : 0.8913
 Specificity : 0.7273
 Pos Pred Value : 0.7736
 Neg Pred Value : 0.8649
 Prevalence : 0.5111
 Detection Rate : 0.4556
 Detection Prevalence : 0.5889
 Balanced Accuracy : 0.8093

'Positive' Class : 0

Alternate answer

Confusion Matrix and Statistics

Reference
 Prediction 0 1
 0 42 13
 1 4 31

Accuracy : 0.8111
 95% CI : (0.7149, 0.8859)
 No Information Rate : 0.5111
 P-Value [Acc > NIR] : 3.351e-09

Kappa : 0.6203

Mcnemar's Test P-Value : 0.05235

Sensitivity : 0.9130
 Specificity : 0.7045
 Pos Pred Value : 0.7636
 Neg Pred Value : 0.8857
 Prevalence : 0.5111
 Detection Rate : 0.4667
 Detection Prevalence : 0.6111
 Balanced Accuracy : 0.8088

'Positive' Class : 0

(4)

vii)

```
tree_accuracy <- confusion_matrix_tree$overall['Accuracy']
rf_accuracy_1 <- confusionMatrix(pred_rf_1, test_data$target)$overall['Accuracy']
rf_accuracy_2 <- confusionMatrix(pred_rf_2, test_data$target)$overall['Accuracy']
cat("Decision Tree Accuracy:", tree_accuracy, "\n")
cat("Random Forest Accuracy mtry 13:", rf_accuracy_1, "\n")
cat("Random Forest Accuracy default mtry:", rf_accuracy_2, "\n")
Decision Tree Accuracy: 0.7666667
Random Forest Accuracy mtry 13: 0.8
Random Forest Accuracy default mtry: 0.8111111
```

The accuracy for the decision tree model is the least, however Random Forest models have better accuracy in this particular scenario.

Among the 2 random forest models, the model with default value of mtry has better accuracy than the model which uses all the parameters.

Using all predictor variables while fitting Random Forest is bagging. While using 1 predictor variable will overfit the model. Here, Default value (13/3 ~ rounded off to 4) chooses the 4 predictor variables at random and fits the model to give the output. Thus, this increases the accuracy and at the same time does not over fits.

(3)

viii)

Decision Trees:

- a) Prone to high variance,
- b) Sensitive to data,
- c) Can easily overfit due to their flexibility and
- d) Ability to capture detailed patterns (including noise) in the training data.

Random Forest:

- a) Combines multiple randomized trees and combines their predictions,
- b) This reduces variance and thus overfitting by diluting the impact of noise and over-specific patterns learned by individual trees.
- c) The randomness in feature selection and bootstrapped data sampling ensures diversity in the trees

(3)

ix)

ROC Curve 3 has the highest AUC of 0.99. This is a measure of model performance.

Also, ROC Curve 3 has strictly higher True Positives rates for false positives rates at 5% indicating high predictive power.

Thus, ROC Curve 3 indicates best model performance among the given three, due to the above stated reasons.

AUC for random classifier is 0.5

AUC for perfect classifier is 1.

(3)

[33 Marks]

Q. 2)

i.

```
install.packages("survival", type = "binary")
install.packages("survminer", type = "binary")
install.packages("ggplot2", type = "binary")
```

```
library(survival)
library(survminer)
library(ggplot2)
```

```
Cruise<- read.csv("<path>/Cruise.csv")
```

```
summary(Cruise$time)
```

```
Min. 1st Qu. Median Mean 3rd Qu. Max.
 2.0 114.2 300.0 203.0 300.0 300.0
```

```
table(Cruise$status)
0 1
105 95
```

(2)

ii.

```
km_fit <- survfit(Surv(time,status) ~ 1, data = Cruise)
```

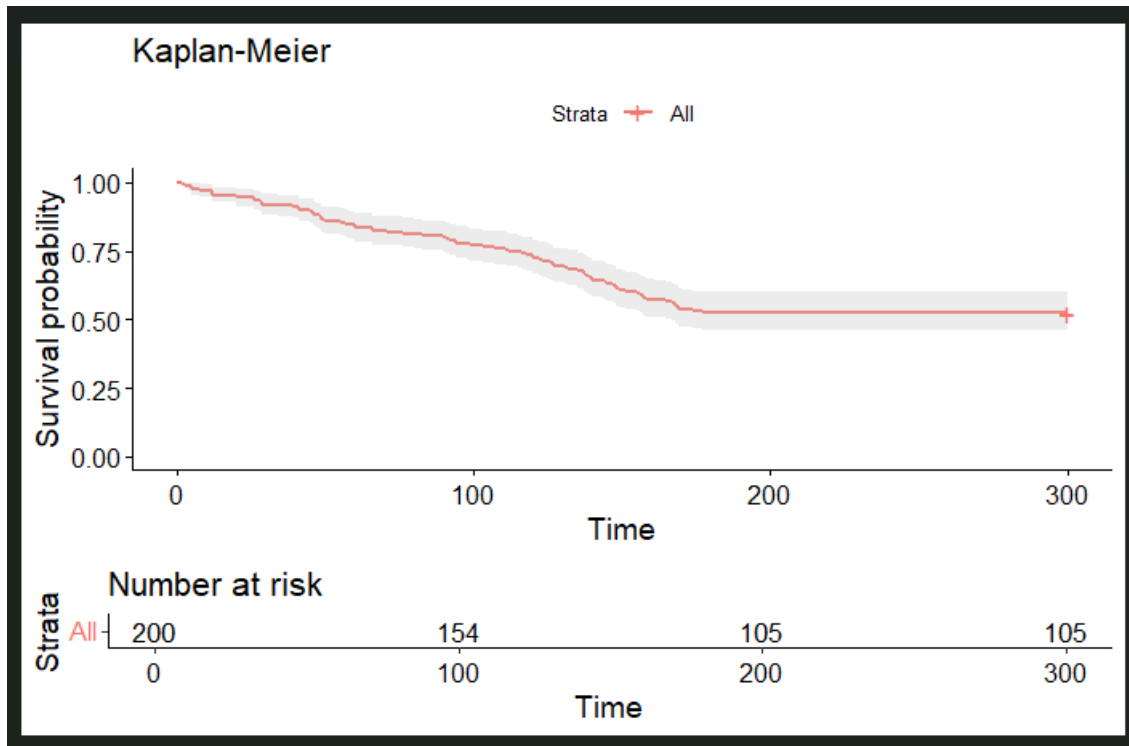
```
summary(km_fit)
```

Call: survfit(formula = Surv(time, status) ~ 1, data = Cruise)

time	n.risk	n.event	survival	std.err	lower	95% CI	upper	95% CI
2	200	1	0.995	0.00499		0.985		1.000
3	199	1	0.990	0.00704		0.976		1.000
4	198	1	0.985	0.00860		0.968		1.000
5	197	2	0.975	0.01104		0.954		0.997
8	195	1	0.970	0.01206		0.947		0.994
12	194	3	0.955	0.01466		0.927		0.984
20	191	2	0.945	0.01612		0.914		0.977
26	189	2	0.935	0.01743		0.901		0.970
28	187	1	0.930	0.01804		0.895		0.966
29	186	2	0.920	0.01918		0.883		0.958
34	184	1	0.915	0.01972		0.877		0.954
39	183	1	0.910	0.02024		0.871		0.951
41	182	2	0.900	0.02121		0.859		0.943
45	180	1	0.895	0.02168		0.854		0.939
46	179	2	0.885	0.02256		0.842		0.930
47	177	1	0.880	0.02298		0.836		0.926
48	176	1	0.875	0.02339		0.830		0.922
49	175	2	0.865	0.02416		0.819		0.914
50	173	1	0.860	0.02454		0.813		0.909
55	172	1	0.855	0.02490		0.808		0.905
57	171	1	0.850	0.02525		0.802		0.901
58	170	1	0.845	0.02559		0.796		0.897
60	169	2	0.835	0.02625		0.785		0.888
66	167	2	0.825	0.02687		0.774		0.879
71	165	1	0.820	0.02717		0.768		0.875
76	164	1	0.815	0.02746		0.763		0.871
80	163	1	0.810	0.02774		0.757		0.866
83	162	1	0.805	0.02802		0.752		0.862
90	161	1	0.800	0.02828		0.746		0.857
91	160	1	0.795	0.02855		0.741		0.853
92	159	1	0.790	0.02880		0.736		0.849
94	158	2	0.780	0.02929		0.725		0.840
98	156	1	0.775	0.02953		0.719		0.835
99	155	1	0.770	0.02976		0.714		0.831
103	154	1	0.765	0.02998		0.708		0.826
107	153	1	0.760	0.03020		0.703		0.822
111	152	1	0.755	0.03041		0.698		0.817
112	151	1	0.750	0.03062		0.692		0.812
115	150	1	0.745	0.03082		0.687		0.808
116	149	1	0.740	0.03102		0.682		0.803
118	148	1	0.735	0.03121		0.676		0.799
120	147	2	0.725	0.03157		0.666		0.790
122	145	1	0.720	0.03175		0.660		0.785
123	144	1	0.715	0.03192		0.655		0.780
125	143	1	0.710	0.03209		0.650		0.776
126	142	1	0.705	0.03225		0.645		0.771
127	141	2	0.695	0.03256		0.634		0.762
131	139	1	0.690	0.03270		0.629		0.757
132	138	1	0.685	0.03285		0.624		0.752
135	137	2	0.675	0.03312		0.613		0.743
137	135	2	0.665	0.03337		0.603		0.734
138	133	1	0.660	0.03350		0.598		0.729
139	132	1	0.655	0.03361		0.592		0.724
140	131	2	0.645	0.03384		0.582		0.715
143	129	1	0.640	0.03394		0.577		0.710
144	128	1	0.635	0.03404		0.572		0.705
145	127	1	0.630	0.03414		0.567		0.701
147	126	1	0.625	0.03423		0.561		0.696
148	125	2	0.615	0.03441		0.551		0.686
149	123	1	0.610	0.03449		0.546		0.681
150	122	1	0.605	0.03457		0.541		0.677
151	121	1	0.600	0.03464		0.536		0.672
155	120	1	0.595	0.03471		0.531		0.667
156	119	1	0.590	0.03478		0.526		0.662
157	118	2	0.580	0.03490		0.515		0.653
158	116	1	0.575	0.03496		0.510		0.648
161	115	1	0.570	0.03501		0.505		0.643
165	114	1	0.565	0.03506		0.500		0.638
167	113	1	0.560	0.03510		0.495		0.633
168	112	2	0.550	0.03518		0.485		0.623
169	110	2	0.540	0.03524		0.475		0.614
170	108	1	0.535	0.03527		0.470		0.609

174	107	1	0.530	0.03529	0.465	0.604
177	106	1	0.525	0.03531	0.460	0.599

```
ggsurvplot(km_fit,data = Cruise,title = "Kaplan-Meier", combine = TRUE, conf.int = TRUE,
risk.table = TRUE)
```



(4)

iii.

```
km_fit_treatment <- survfit(Surv(time, status) ~ Pclass, data = Cruise)
```

```
summary(km_fit_treatment)
```

```
Call: survfit(formula = Surv(time, status) ~ Pclass, data = Cruise)
```

```

Pclass=1
time n.risk n.event survival std.err lower 95% CI upper 95% CI
 2   65    1  0.985 0.0153   0.955   1.000
 8   64    1  0.969 0.0214   0.928   1.000
34   63    1  0.954 0.0260   0.904   1.000
39   62    1  0.938 0.0298   0.882   0.999
58   61    1  0.923 0.0331   0.861   0.990
60   60    1  0.908 0.0359   0.840   0.981
71   59    1  0.892 0.0384   0.820   0.971
94   58    1  0.877 0.0407   0.801   0.961
99   57    1  0.862 0.0428   0.782   0.950
107  56    1  0.846 0.0448   0.763   0.939
111  55    1  0.831 0.0465   0.744   0.927
135  54    1  0.815 0.0481   0.726   0.915
140  53    1  0.800 0.0496   0.708   0.903
155  52    1  0.785 0.0510   0.691   0.891
161  51    1  0.769 0.0523   0.673   0.879
165  50    1  0.754 0.0534   0.656   0.866
169  49    1  0.738 0.0545   0.639   0.853
177  48    1  0.723 0.0555   0.622   0.840

```

Pclass=2

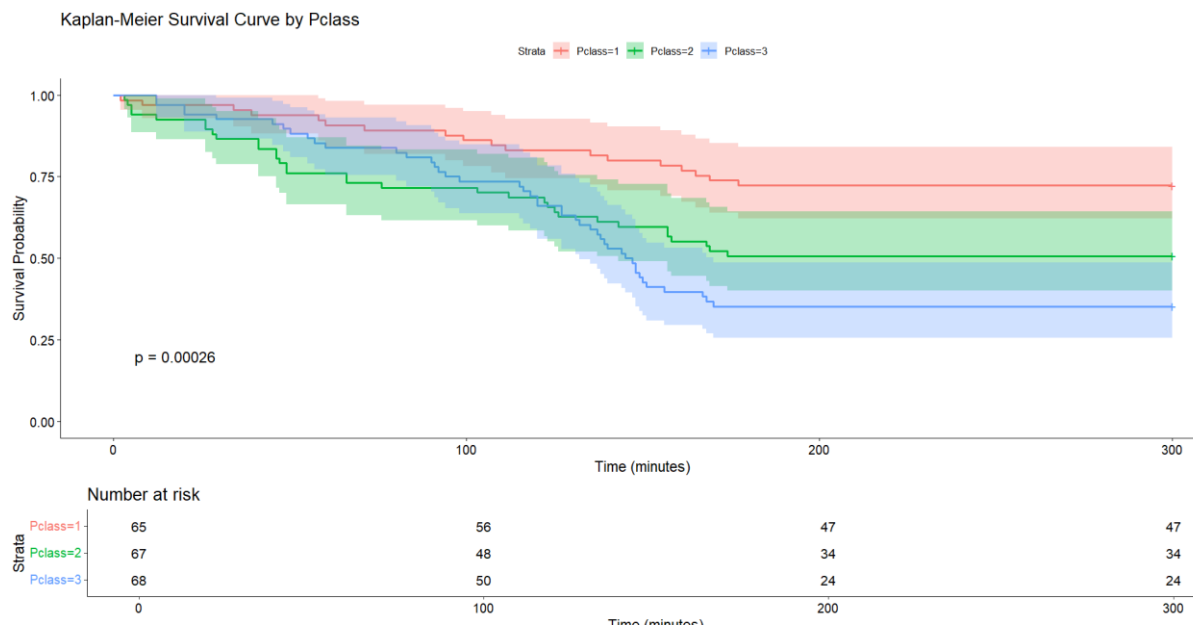
time	n.risk	n.event	survival	std.err	lower 95% CI	upper 95% CI
3	67	1	0.985	0.0148	0.956	1.000
4	66	1	0.970	0.0208	0.930	1.000
5	65	2	0.940	0.0289	0.885	0.999
12	63	1	0.925	0.0321	0.865	0.990
26	62	2	0.896	0.0374	0.825	0.972
28	60	1	0.881	0.0396	0.806	0.962
29	59	1	0.866	0.0417	0.788	0.951
41	58	2	0.836	0.0453	0.752	0.929
46	56	2	0.806	0.0483	0.717	0.906
47	54	1	0.791	0.0497	0.699	0.895
49	53	2	0.761	0.0521	0.666	0.870
66	51	2	0.731	0.0542	0.633	0.846
76	49	1	0.716	0.0551	0.616	0.833
103	48	1	0.701	0.0559	0.600	0.820
112	47	1	0.687	0.0567	0.584	0.807
122	46	1	0.672	0.0574	0.568	0.794
123	45	1	0.657	0.0580	0.552	0.781
125	44	1	0.642	0.0586	0.537	0.768
126	43	1	0.627	0.0591	0.521	0.754
137	42	1	0.612	0.0595	0.506	0.740
143	41	1	0.597	0.0599	0.490	0.727
157	40	2	0.567	0.0605	0.460	0.699
158	38	1	0.552	0.0608	0.445	0.685
168	37	1	0.537	0.0609	0.430	0.671
169	36	1	0.522	0.0610	0.415	0.657
174	35	1	0.507	0.0611	0.401	0.642

Pclass=3

time	n.risk	n.event	survival	std.err	lower 95% CI	upper 95% CI
12	68	2	0.971	0.0205	0.931	1.000
20	66	2	0.941	0.0285	0.887	0.999
29	64	1	0.926	0.0317	0.866	0.991
45	63	1	0.912	0.0344	0.847	0.982
48	62	1	0.897	0.0369	0.828	0.972
50	61	1	0.882	0.0391	0.809	0.962
55	60	1	0.868	0.0411	0.791	0.952
57	59	1	0.853	0.0429	0.773	0.941
60	58	1	0.838	0.0447	0.755	0.930
80	57	1	0.824	0.0462	0.738	0.919
83	56	1	0.809	0.0477	0.721	0.908
90	55	1	0.794	0.0490	0.704	0.896
91	54	1	0.779	0.0503	0.687	0.884
92	53	1	0.765	0.0514	0.670	0.872
94	52	1	0.750	0.0525	0.654	0.860
98	51	1	0.735	0.0535	0.638	0.848
115	50	1	0.721	0.0544	0.621	0.836
116	49	1	0.706	0.0553	0.605	0.823
118	48	1	0.691	0.0560	0.590	0.810
120	47	2	0.662	0.0574	0.558	0.784
127	45	2	0.632	0.0585	0.528	0.758
131	43	1	0.618	0.0589	0.512	0.745

132	42	1	0.603	0.0593	0.497	0.731
135	41	1	0.588	0.0597	0.482	0.718
137	40	1	0.574	0.0600	0.467	0.704
138	39	1	0.559	0.0602	0.452	0.690
139	38	1	0.544	0.0604	0.438	0.676
140	37	1	0.529	0.0605	0.423	0.662
144	36	1	0.515	0.0606	0.409	0.648
145	35	1	0.500	0.0606	0.394	0.634
147	34	1	0.485	0.0606	0.380	0.620
148	33	2	0.456	0.0604	0.352	0.591
149	31	1	0.441	0.0602	0.338	0.576
150	30	1	0.426	0.0600	0.324	0.562
151	29	1	0.412	0.0597	0.310	0.547
156	28	1	0.397	0.0593	0.296	0.532
167	27	1	0.382	0.0589	0.283	0.517
168	26	1	0.368	0.0585	0.269	0.502
170	25	1	0.353	0.0580	0.256	0.487

```
ggsurvplot(km_fit_treatment, data = Cruise, conf.int = TRUE, xlab = "Time
(minutes)", ylab = "Survival Probability", title = "Kaplan-Meier Survival
Curve by Pclass", pval = TRUE, risk.table = TRUE)
```



Passenger travelling in Class 1 had the highest survival probability followed by Passenger travelling in Class 2 & 3.

(4)

iv.

```
cox_fit <- coxph(Surv(time, status) ~ Age + Sex + Pclass + Mstatus + Parch + SibSp +
Embarked, data = Cruise)
```

```
summary(cox_fit)
```

Call:

```
coxph(formula = Surv(time, status) ~ Age + Sex + Pclass + Mstatus +
Parch + SibSp + Embarked, data = Cruise)
```

n= 200, number of events= 95

	coef	exp(coef)	se(coef)	z	Pr(> z)
Age	0.002217	1.002219	0.005214	0.425	0.6707
Sex	0.477555	1.612128	0.217117	2.200	0.0278 *
Pclass	0.564965	1.759386	0.135786	4.161	3.17e-05 ***
Mstatus	-0.010950	0.989110	0.105248	-0.104	0.9171
Parch	-0.212472	0.808583	0.106457	-1.996	0.0460 *
SibSp	0.181520	1.199038	0.132582	1.369	0.1710
Embarked	-0.200554	0.818277	0.128254	-1.564	0.1179

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

	exp(coef)	exp(-coef)	lower .95	upper .95
Age	1.0022	0.9978	0.9920	1.0125
Sex	1.6121	0.6203	1.0534	2.4672
Pclass	1.7594	0.5684	1.3483	2.2958
Mstatus	0.9891	1.0110	0.8047	1.2157
Parch	0.8086	1.2367	0.6563	0.9962
SibSp	1.1990	0.8340	0.9247	1.5548
Embarked	0.8183	1.2221	0.6364	1.0521

Concordance= 0.646 (se = 0.028)
Likelihood ratio test= 28.77 on 7 df, p=2e-04
Wald test = 27.36 on 7 df, p=3e-04
Score (logrank) test = 28.67 on 7 df, p=2e-04

(3)

v.

(a) The **coef** column represents the **log hazard ratios** (log(HR)). These coefficients tell us how much each unit increase in the predictor variable affects the hazard rate (risk of the event). A positive coefficient indicates that an increase in the predictor leads to an increased hazard (higher risk), while a negative coefficient suggests a decreased hazard (lower risk).

Age: The coefficient is 0.002217, suggesting a very slight increase in hazard with age.

Sex: The coefficient is 0.477555, meaning being male (since male is coded as 1 and female as 0) increases the hazard compared to being female.

Pclass: The coefficient is 0.564965, indicating a higher passenger class (i.e. Passenger class 2 & 3) is associated with a higher hazard as compared to Passenger class 1.

Mstatus: The coefficient is -0.010950, indicating as the factor increases and moves towards 3 i.e. Married the risk reduces marginally.

Parch: The coefficient is -0.212472, indicating as the number of parents/children aboard the ship increases the risk reduces sharply.

SibSp: The coefficient is 0.181520, indicating as the spouse/higher number of siblings being aboard the ship increases the risk reduces increases.

Embarked: The coefficient is -0.200554, indicating the risk decreases for passengers boarding last.

(b) **Small p-values (usually < 0.05) suggest that the predictor is statistically significant in the model.**

Thus, Sex, Parch & Pclass are the most statistically significant as their p values are less than 0.05.

(c) The **lower .95** and **upper .95** columns show the 95% confidence interval for the hazard ratio. If this interval excludes 1, it indicates that the predictor likely has a significant effect.

Sex, Pclass and Parch excludes the confidence interval 1 and thus are the significant effect in line with our conclusion in previous sub part.

(7)

vi.

```
exp(coef(cox_fit))
```

Age	Sex	Pclass	Mstatus	Parch	SibSp	Embarked
1.0022192	1.6121277	1.7593859	0.9891096	0.8085829	1.1990382	0.8182769

The **exp(coef)** column shows the **hazard ratios (HR)**, which are easier to interpret than the raw coefficients. The hazard ratio tells us the relative risk of the event per unit increase in the predictor.

```
exp(coef)
```

```
Age    1.0022
```

```
Sex    1.6121
```

```
Pclass 1.7594
```

```
Mstatus 0.9891
```

```
Parch  0.8086
```

```
SibSp  1.1990
```

```
Embarked 0.8183
```

Age (HR = 1.0022): For each additional year of age, the hazard (risk of event) increases by 0.22%.

Sex (HR = 1.6121): Males have a 61.2% higher risk of the death compared to females.

Pclass (HR = 1.7594): For each unit increase in the Pclass factor i.e. Pclass changing from 1 to 2, the hazard increases by 75.94%.

Mstatus (HR = 0.9891): For each unit increase in the Mstatus factor i.e. Mstatus changing from 0 to 1, the hazard decreases by 1.09%.

Parch (HR = 0.8086): For each unit increase in the number of parents/children of the passenger being aboard, the hazard decreases by 19.14%.

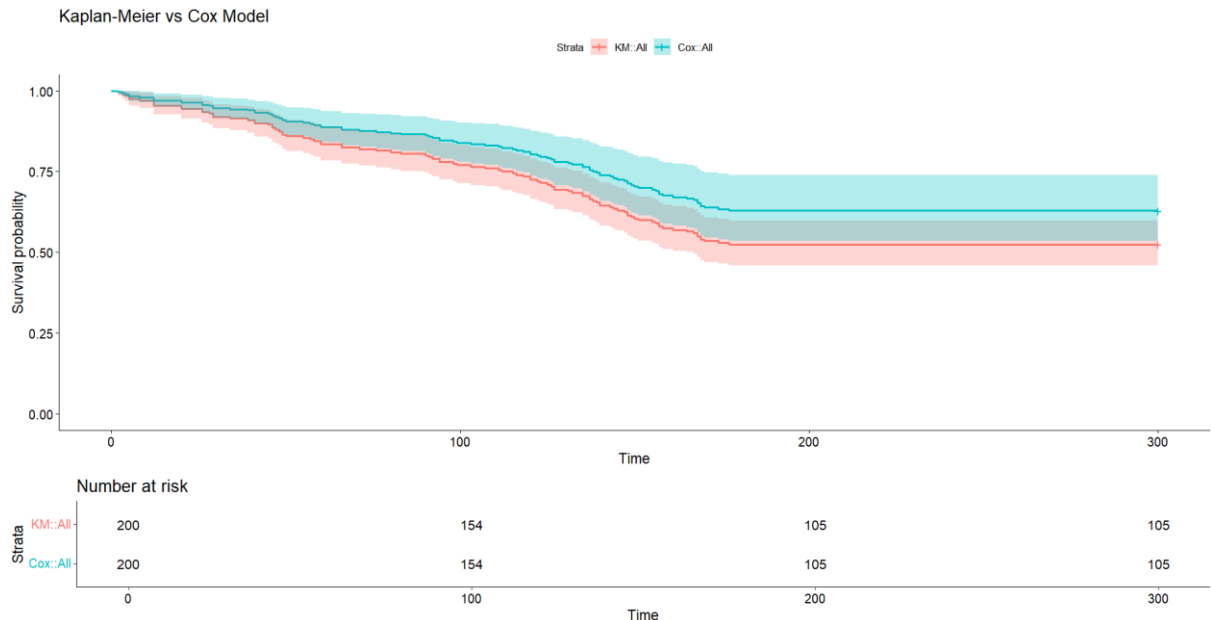
SibSp (HR = 1.1990): For each unit increase in the number of siblings or spouse travelling with the passenger being aboard, the hazard increases by 19.90%.

Embarked (HR = 0.8183): For each unit increase in the Embarking station factor, the hazard decreases by 18.17%. Since the embarking station has been numbered in a manner that last embarking station has the highest factor, we can say that as with each subsequent embarking station the risk decreases by 18.17%.

(4)

vii.

```
ggsurvplot(list(KM = km_fit, Cox = survfit(cox_fit)), data = Cruise, title = "Kaplan-Meier vs Cox Model", combine = TRUE, conf.int = TRUE, pval = TRUE, risk.table = TRUE)
```



(2)

viii.

Use Kaplan-Meier When:

- You want to **visualize and describe** survival times between two or more groups.
- You have a simple comparison between groups based on one categorical variable.
- You're working with **small datasets** or want a simple descriptive analysis.
- The focus is on **estimating survival probabilities** over time without adjusting for other factors.

Use CoxPH When:

- You need to **adjust for multiple covariates** (e.g., age, treatment, gender) to assess their independent effect on survival.
- You want to estimate **hazard ratios** to quantify the relative risk associated with covariates, as calculated in (iv).
- You're building a **predictive model** to estimate survival times based on individual characteristics.
- You're interested in testing **interactions** between covariates.
- The dataset contains **complex factors** that need to be considered simultaneously.

(2)

[28 Marks]

Q. 3)

i.

```
install.packages("markovchain", type = "binary")
install.packages("dplyr", type = "binary")
```

```
library(markovchain)
library(dplyr)
```

```
MC <- new("markovchain", states = c("1","2","3"),
         transitionMatrix = matrix(data = c(0.1,0.6,0.3,0.4,0.1,0.5,0.2,0.3,0.5),
                                   byrow = TRUE , nrow = 3), name = "MC")
```

MC

A 3 - dimensional discrete Markov Chain defined by the following states:

1, 2, 3

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

	1	2	3
1	0.1	0.6	0.3
2	0.4	0.1	0.5
3	0.2	0.3	0.5

(2)

ii.

```
markovchain::summary(MC)
```

MC Markov chain that is composed by:

Closed classes:

1 2 3

Recurrent classes:

{1,2,3}

Transient classes:

NONE

The Markov chain is irreducible

The absorbing states are: NONE

Alternate Answer

```
markovchain::is.irreducible(MC)
```

TRUE

(1)

iii.

```
markovchain::meanFirstPassageTime(MC)
```

	1	2	3
1	0.000000	2.051282	2.631579
2	3.333333	0.000000	2.280702
3	4.000000	2.820513	0.000000

```
markovchain::meanRecurrenceTime(MC)
```

	1	2	3
1	4.200000	3.230769	2.210526

(4)

iv.

```
InitialState <- c(1,0,0)
```

```
After2steps <- InitialState * (MC^2)
```

```
After4steps <- InitialState * (MC^4)
```

```
After6steps <- InitialState * (MC^6)
```

```

>After2steps
  1  2  3
[1,] 0.31 0.21 0.48

> After4steps
  1  2  3
[1,] 0.2491 0.2931 0.4578

> After6steps
  1  2  3
[1,] 0.239851 0.306891 0.453258

>steadyStates(MC)
  1  2  3
[1,] 0.2380952 0.3095238 0.452381

```

(4)

v.

```

stat_dist <- data.frame(c(After2steps),c(After4steps),c(After6steps),c(steadyStates(MC)))
stat_dist

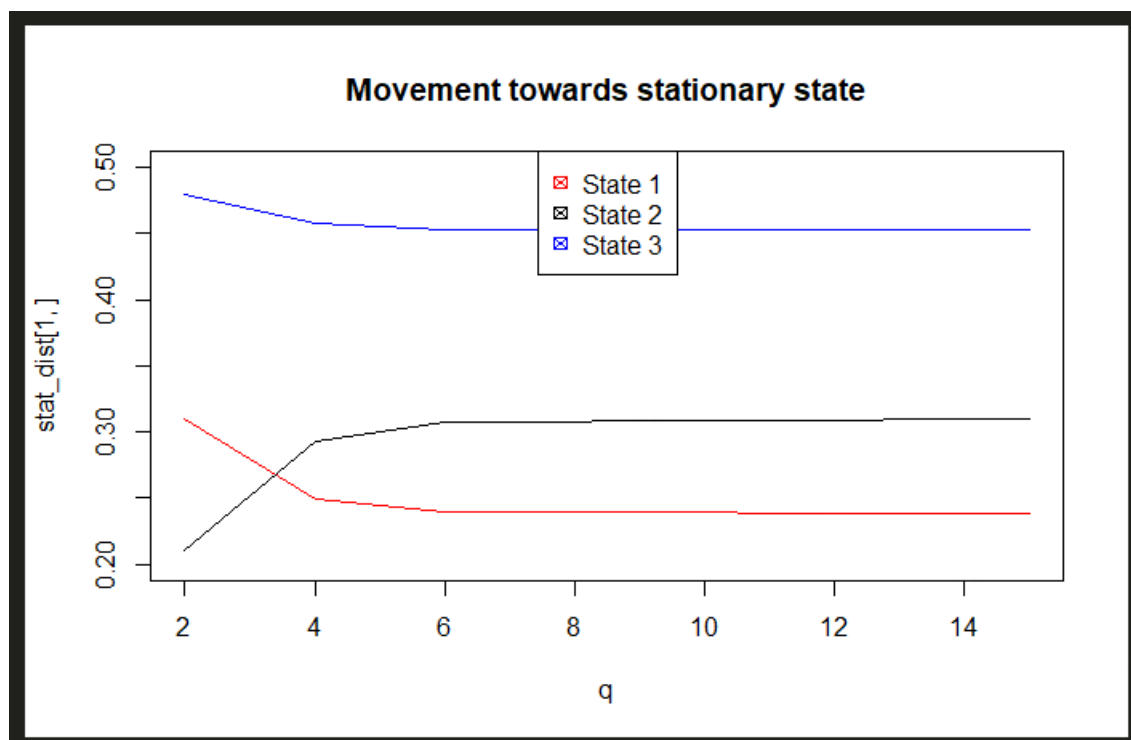
```

```
q = c(2,4,6,15)
```

```

plot(q,stat_dist[1,],type = "l",main = "Movement towards stationary state",col = "red", ylim =
c(0.2,0.5))
lines(q,stat_dist[2,], col = "black")
lines(q,stat_dist[3,], col = "blue")
legend("top",legend = c("State 1", "State 2", "State 3"),col = c("red", "black", "blue"),
      pch = 7)

```



After 6 steps the Markov chain almost becomes stationary.

(6)

```

vi.
set.seed(42)
n_steps <- 10000
states <- numeric(n_steps)
states[1] <- 1
for (i in 2:n_steps) {
  states[i] <- sample(1:3, 1, prob = MC[states[i-1], ])
}

```

```
head(states)
```

```
[1] 1 1 1 2 1 3
```

(4)

```

vii.
triplet_counts <- c(
  sum(states[-c(1, 2)] == 1 & states[-length(states)] == 1 & states[-1] == 1),
  sum(states[-c(1, 2)] == 2 & states[-length(states)] == 1 & states[-1] == 1),
  sum(states[-c(1, 2)] == 3 & states[-length(states)] == 1 & states[-1] == 1),
  sum(states[-c(1, 2)] == 1 & states[-length(states)] == 2 & states[-1] == 1),
  sum(states[-c(1, 2)] == 2 & states[-length(states)] == 2 & states[-1] == 1),
  sum(states[-c(1, 2)] == 3 & states[-length(states)] == 2 & states[-1] == 1),
  sum(states[-c(1, 2)] == 1 & states[-length(states)] == 3 & states[-1] == 1),
  sum(states[-c(1, 2)] == 2 & states[-length(states)] == 3 & states[-1] == 1),
  sum(states[-c(1, 2)] == 3 & states[-length(states)] == 3 & states[-1] == 1),
  sum(states[-c(1, 2)] == 1 & states[-length(states)] == 1 & states[-1] == 2),
  sum(states[-c(1, 2)] == 2 & states[-length(states)] == 1 & states[-1] == 2),
  sum(states[-c(1, 2)] == 3 & states[-length(states)] == 1 & states[-1] == 2),
  sum(states[-c(1, 2)] == 1 & states[-length(states)] == 2 & states[-1] == 2),
  sum(states[-c(1, 2)] == 2 & states[-length(states)] == 2 & states[-1] == 2),
  sum(states[-c(1, 2)] == 3 & states[-length(states)] == 2 & states[-1] == 2),
  sum(states[-c(1, 2)] == 1 & states[-length(states)] == 3 & states[-1] == 2),
  sum(states[-c(1, 2)] == 2 & states[-length(states)] == 3 & states[-1] == 2),
  sum(states[-c(1, 2)] == 3 & states[-length(states)] == 3 & states[-1] == 2),
  sum(states[-c(1, 2)] == 1 & states[-length(states)] == 1 & states[-1] == 3),
  sum(states[-c(1, 2)] == 2 & states[-length(states)] == 1 & states[-1] == 3),
  sum(states[-c(1, 2)] == 3 & states[-length(states)] == 1 & states[-1] == 3),
  sum(states[-c(1, 2)] == 1 & states[-length(states)] == 2 & states[-1] == 3),
  sum(states[-c(1, 2)] == 2 & states[-length(states)] == 2 & states[-1] == 3),
  sum(states[-c(1, 2)] == 3 & states[-length(states)] == 2 & states[-1] == 3),
  sum(states[-c(1, 2)] == 1 & states[-length(states)] == 3 & states[-1] == 3),
  sum(states[-c(1, 2)] == 2 & states[-length(states)] == 3 & states[-1] == 3),
  sum(states[-c(1, 2)] == 3 & states[-length(states)] == 3 & states[-1] == 3)
)

```

```

ijk <-
c(111,112,113,121,122,123,131,132,133,211,212,213,221,222,223,231,232,233,311,312,313
,321,322,323,331,332,333)
triplets_test <- data.frame(ijk,triplet_counts)
triplets_test

```

```

ijk triplet_counts
1 111      30
2 112     158
3 113      69
4 121     111
5 122     787
6 123     346
7 131     115
8 132     520
9 133     275
10 211     582
11 212     151
12 213     732
13 221     117
14 222      25
15 223     174
16 231     545
17 232     140
18 233     648
19 311     132
20 312     207
21 313     351
22 321     297
23 322     487
24 323     770
25 331     482
26 332     639
27 333    1109

```

(6)

viii.

```

double_counts <- c(
  sum(states[-length(states)] == 1 & states[-1] == 1),
  sum(states[-length(states)] == 2 & states[-1] == 1),
  sum(states[-length(states)] == 3 & states[-1] == 1),
  sum(states[-length(states)] == 1 & states[-1] == 2),
  sum(states[-length(states)] == 2 & states[-1] == 2),
  sum(states[-length(states)] == 3 & states[-1] == 2),
  sum(states[-length(states)] == 1 & states[-1] == 3),
  sum(states[-length(states)] == 2 & states[-1] == 3),
  sum(states[-length(states)] == 3 & states[-1] == 3)
)

```

```

ij <- c(11,12,13,21,22,23,31,32,33)
double_counts <- data.frame(ij,double_counts)
double_counts$ij <- as.character(double_counts$ij)
double_counts
  ij double_counts
1 11      257
2 12     1244
3 13      910
4 21     1465
5 22      316

```

```
6 23    1333
7 31    690
8 32    1554
9 33    2230
```

(2)

ix.

```
triplets_test$ij <- substr(triplets_test$ijk,1,2)
merged_df <- left_join(triplets_test,double_counts, by = "ij")

tp <- c(0.1,0.6,0.3,0.4,0.1,0.5,0.2,0.3,0.5)
jk <- c(11,12,13,21,22,23,31,32,33)

t_prob <- data.frame(jk,tp)
t_prob$jk <- as.character(t_prob$jk)

merged_df$jk <- substr(triplets_test$ijk,2,3)

merged_df1 <- left_join(merged_df,t_prob, by = "jk")

merged_df1$ef <- merged_df1$double_counts*merged_df1$tp

merged_df1$ts <- (((merged_df1$triplet_counts - merged_df1$ef)^2)/merged_df1$ef)

chi_square_test <- chisq.test(merged_df1$triplet_counts,p =
merged_df1$ef/sum(merged_df1$ef))
chi_square_test

#Chi-squared test for given probabilities

data: merged_df1$triplet_counts
X-squared = 7971.8, df = 26, p-value < 2.2e-16

if (chi_square_test$p.value < 0.05) {
  cat("Reject the null hypothesis: The observed frequencies do not match the expected
frequencies.\n")
} else {
  cat("Fail to reject the null hypothesis: The observed frequencies match the expected
frequencies.\n")
}
```

Reject the null hypothesis: The observed frequencies do not match the expected frequencies.

(10)

[39 Marks]