ECM2002 Machine Learning Algorithms

Submitted by

NAME: Tarunveer Singh Sidhu

Register Number:19BLC1186

Laboratory Component

in partial fulfilment for the award of the degree of

BACHELOR OF TECHNOLOGY

in

ELECTRONICS AND COMPUTER ENGINEERING

VIT®
Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

February - May 2021

TABLE OF CONTENTS

Lab Exercise No : 1
Exercise Title : Datasets and R Programming
Date : 08-02-2021

Task:
To use R script to manipulate the given dataset

Program:

```
library(ISLR)
data(Carseats)
summary(Carseats)
names(Carseats)
fix
data(Carseats)

str(Carseats)
install.packages("e1071")
install.packages("caTools")
install.packages("class")

library(e1071)
library(caTools)
library(class)

data(Carseats)
head(Carseats)

split <- sample.split(Carseats, SplitRatio = 0.9)
train_cl <- subset(Carseats, split == "TRUE")
test_cl <- subset(Carseats, split == "FALSE")

train_scale <- scale(train_cl[, 1:4])
test_scale <- scale(test_cl[, 1:4])
train_scale

classifier_knn <- knn(train = train_scale,
                      test = test_scale,
                      cl = train_cl$Urban,
                      k = 1)
classifier_knn

cm <- table(test_cl$Urban, classifier_knn)
cm

misClassError <- mean(classifier_knn != test_cl$Urban)
print(paste('Accuracy =', 1-misClassError))

accuracies <- vector()

for(i in 1:20){
  print(paste("For k = ",i))
  classifier_knn <- knn(train = train_scale,
                        test = test_scale,
                        cl = train_cl$Urban,
                        k = i)
  misClassError <- mean(classifier_knn != test_cl$Urban)
  print(paste('Accuracy =', 1-misClassError))
  accuracies[i] <- 1-misClassError
}
print(accuracies)
print(1:20)
plot(1:20,accuracies,ylab="Accuracy",xlab="K value", type='l')
```

Output:

```
Output:
> library(ISLR)

Attaching package: ‘ISLR’

The following object is masked _by_ ‘.GlobalEnv’:

    Auto

> data(Carseats)
> summary(Carseats)
     Sales          CompPrice  
 Min.   : 0.000   Min.   : 77  
 1st Qu.: 5.390   1st Qu.:115  
 Median : 7.490   Median :125  
 Mean   : 7.496   Mean   :125  
 3rd Qu.: 9.320   3rd Qu.:135  
 Max.   :16.270   Max.   :175  
     Income        Advertising    
 Min.   : 21.00   Min.   : 0.000  
 1st Qu.: 42.75   1st Qu.: 0.000  
 Median : 69.00   Median : 5.000  
 Mean   : 68.66   Mean   : 6.635  
 3rd Qu.: 91.00   3rd Qu.:12.000  
 Max.   :120.00   Max.   :29.000  
   Population        Price        ShelveLoc  
 Min.   : 10.0   Min.   : 24.0   Bad   : 96  
 1st Qu.:139.0   1st Qu.:100.0   Good  : 85  
 Median :272.0   Median :117.0   Medium:219  
 Mean   :264.8   Mean   :115.8               
 3rd Qu.:398.5   3rd Qu.:131.0               
 Max.   :509.0   Max.   :191.0               
      Age          Education    Urban    
 Min.   :25.00   Min.   :10.0   No :118  
 1st Qu.:39.75   1st Qu.:12.0   Yes:282  
 Median :54.50   Median :14.0            
 Mean   :53.32   Mean   :13.9            
 3rd Qu.:66.00   3rd Qu.:16.0            
 Max.   :80.00   Max.   :18.0            
   US     
 No :142  
 Yes:258  
```

```
> names(Carseats)
 [1] "Sales"       "CompPrice"   "Income"
 [4] "Advertising" "Population"  "Price"
 [7] "ShelveLoc"   "Age"         "Education"
[10] "Urban"       "US"
> fix
function (x, ...)
{
    subx <- substitute(x)
    if (is.name(subx))
        subx <- deparse(subx)
    if (!is.character(subx) || length(subx) != 1L)
        stop("'fix' requires a name")
    parent <- parent.frame()
    if (exists(subx, envir = parent, inherits = TRUE))
        x <- edit(get(subx, envir = parent), title = subx, ...)
    else {
        x <- edit(function() {
        }, title = subx, ...)
        environment(x) <- .GlobalEnv
    }
    assign(subx, x, envir = .GlobalEnv)
}
<bytecode: 0x7f87594f91c8>
<environment: namespace:utils>
> data(Carseats)
>
> str(Carseats)
'data.frame':	400 obs. of  11 variables:
 $ Sales      : num  9.5 11.22 10.06 7.4 4.15 ...
 $ CompPrice  : num  138 111 113 117 141 124 115 136 132 132 ...
 $ Income     : num  73 48 35 100 64 113 105 81 110 113 ...
 $ Advertising: num  11 16 10 4 3 13 0 15 0 0 ...
 $ Population : num  276 260 269 466 340 501 45 425 108 131 ...
 $ Price      : num  120 83 80 97 128 72 108 120 124 124 ...
 $ ShelveLoc  : Factor w/ 3 levels "Bad","Good","Medium": 1 2 3 3 1 1 3 2 3 3 ...
 $ Age        : num  42 65 59 55 38 78 71 67 76 76 ...
 $ Education  : num  17 10 12 14 13 16 15 10 10 17 ...
 $ Urban      : Factor w/ 2 levels "No","Yes": 2 2 2 2 2 1 2 2 1 1 ...
 $ US         : Factor w/ 2 levels "No","Yes": 2 2 2 2 1 2 1 2 1 2 ...
>
>
>
> library(e1071)
> library(caTools)
> library(class)
>
> data(Carseats)
> head(Carseats)
  Sales CompPrice Income Advertising
1  9.50       138     73          11
2 11.22       111     48          16
3 10.06       113     35          10
4  7.40       117    100           4
5  4.15       141     64           3
```

```
6 10.81       124   113         13
  Population Price ShelveLoc Age Education
1        276   120       Bad  42        17
2        260    83      Good  65        10
3        269    80    Medium  59        12
4        466    97    Medium  55        14
5        340   128       Bad  38        13
6        501    72       Bad  78        16
  Urban  US
1   Yes Yes
2   Yes Yes
3   Yes Yes
4   Yes Yes
5   Yes  No
6    No Yes
>
> split <- sample.split(Carseats, SplitRatio = 0.9)
> train_cl <- subset(Carseats, split == "TRUE")
> test_cl <- subset(Carseats, split == "FALSE")
>
> train_scale <- scale(train_cl[, 1:4])
> test_scale <- scale(test_cl[, 1:4])
> train_scale
          Sales    CompPrice      Income
1    0.695927718  0.832219591  0.15631266
2    1.320284987 -0.905610331 -0.75273970
3    0.899206829 -0.776882188 -1.22544694
4   -0.066368948 -0.519425904  1.13808922
5   -1.246113787  1.025311805 -0.17094619
7   -0.345877725 -0.648154046  1.31989969
9   -0.378547582  0.446035164  1.50171017
10  -1.050094645  0.446035164  1.61079645
11   0.518058496 -0.261969619  0.33812314
12   1.588903813 -0.519425904  0.91991665
13  -1.307823518 -0.197605548 -1.22544694
14   1.225905400 -0.648154046 -1.47998160
15   1.302135067 -1.163066616  1.75624483
16   0.409158973  1.540224374  0.95627875
18   1.708693289  1.411496232  0.19267476
20   0.416418941  0.252942950  0.26539895
21  -0.425737376 -0.004513334  0.77446827
22   1.650613543  0.574763306 -1.44361950
23  -0.908525264  0.188578879 -0.82546389
24  -0.621756518 -0.261969619 -1.37089531
25   0.928246702  1.282768089  1.82896902
26   2.656119145  0.896583662 -1.33453322
27   0.271219576 -1.163066616  1.68352064
29  -1.667191946 -1.420522900  0.19267476
31   2.166071288 -0.004513334  0.91991665
32   0.242179703  0.703491449 -0.38911876
33  -0.501967042 -1.163066616 -1.33453322
34   0.430938877 -0.712518117 -1.11636065
35  -1.783351438 -0.648154046 -0.53456714
36   1.265835226  0.381671093  0.55629571
37   0.474498687 -0.197605548  0.26539895
38  -0.955715058 -0.261969619 -1.00727437
40  -1.576442343  0.317307022 -0.31639457
42   0.136910163  2.055136944 -0.57092923
```

43	1.033516242	-3.093988751	0.01086428
44	-1.257003740	-0.133241477	-0.97091227
45	-1.242483803	-2.579076182	0.37448523
46	-1.097284438	1.025311805	-0.20730828
47	1.763143050	0.124214808	0.77446827
48	-1.162624153	0.059850737	1.06536503
49	-1.333233406	-0.583789975	-0.60729133
51	-2.237099453	-1.677979185	-1.33453322
53	0.118760243	1.797680659	-1.04363646
54	-0.240608185	-1.034338473	-0.17094619
55	-0.973864978	0.574763306	1.24717550
56	-0.266018074	1.154039947	0.44720942
57	1.570753892	0.510399235	0.48357152
58	-2.422228643	-2.064163612	0.81083037
59	-0.785105804	-1.420522900	0.88355456
60	-0.861335470	-0.455061833	0.08358847
62	-0.095408820	-1.291794758	-1.33453322
64	0.322039354	-0.390697761	0.70174408
65	0.078830417	-1.613615114	-0.06185991
66	-0.973864978	-0.197605548	-1.55270579
67	0.459978750	0.124214808	0.84719246
68	0.518058496	0.059850737	-0.28003247
69	2.107991542	1.540224374	0.01086428
70	0.147800116	0.124214808	-0.35275666
71	0.681407782	-2.321619897	0.44720942
73	-0.748805963	-0.648154046	-0.86182599
75	-0.501967042	1.604588445	-0.02549781
76	0.351079227	-2.385983968	1.53807226
77	1.109745908	-1.484886972	0.66538199
78	0.042530576	-0.455061833	0.08358847
79	-1.144474232	0.574763306	-0.75273970
80	0.565248290	0.574763306	-0.06185991
81	0.155060084	-0.776882188	1.13808922
82	-0.022809138	-0.583789975	0.11995057
84	-1.148104216	-1.034338473	-1.18908484
86	0.322039354	-0.004513334	1.24717550
87	0.405528988	1.604588445	0.55629571
88	1.494524225	0.381671093	-0.06185991
89	-0.371287614	-0.519425904	-0.97091227
90	0.133280179	0.188578879	-0.09822200
91	-0.817775661	-0.648154046	-1.69815417
92	-1.006534835	-1.806707327	-0.82546389
93	-1.108174391	-0.712518117	1.61079645
95	0.292999481	-0.648154046	1.02900294
97	0.688667750	1.411496232	-0.97091227
98	-0.048219027	2.312593228	0.48357152
99	1.781292971	-0.197605548	0.30176104
100	-0.981124946	-0.261969619	-0.78910180
101	-1.260633724	-0.776882188	0.01086428
102	-0.501967042	0.188578879	0.88355456
103	-0.828665613	-0.776882188	-1.69815417
104	-0.912155248	-0.133241477	0.81083037
106	-0.737916010	-1.356158829	1.13808922
108	0.351079227	0.574763306	1.39262388
109	-1.492952708	-1.163066616	0.37448523
110	0.507168544	-0.648154046	-0.13458409
111	0.514428512	0.188578879	-0.24367038
112	-0.349507709	0.446035164	1.79260693

```
113 -0.331357788 -0.583789975  1.10172713
114 -0.570936741  0.381671093 -1.44361950
115  0.626958020 -0.197605548  0.66538199
117 -0.908525264  0.639127378  0.22903685
119 -0.004659217 -0.841246260  0.70174408
120 -0.077258900  0.317307022  0.91991665
121 -0.258758106  0.188578879  1.31989969
122  1.483634273 -0.004513334  0.73810618
123 -0.255128122 -0.390697761  1.13808922
124  0.220399798  0.124214808  1.24717550
125  0.467238719  0.381671093  1.61079645
126  0.637847972 -2.321619897  0.33812314
128 -0.385807550 -0.004513334 -0.75273970
130 -1.129954296  1.154039947  1.86533111
131  0.300259449 -1.999799541  0.55629571
132 -0.393067519 -1.098702544  0.01086428
133  0.710447655 -0.004513334  0.66538199
134  0.013490703  0.446035164  1.06536503
135 -1.420353025  0.446035164 -1.37089531
136 -0.414847423 -1.871071399  0.91991665
137 -0.875855407  0.381671093  0.22903685
139  0.975436496 -0.004513334  1.24717550
141 -0.563676772  0.510399235 -0.31639457
142 -0.382177566  0.960947734 -0.97091227
143 -0.051849011 -0.068877405  0.55629571
144 -2.560168040 -0.197605548  0.70174408
145  0.547098369  0.446035164 -0.02549781
146  0.430938877  1.218404018 -0.20730828
147 -1.336863391 -0.712518117  0.51993361
148  1.062556115  0.960947734 -0.53456714
150  1.414664575 -0.261969619  1.86533111
152  1.156935702 -0.905610331 -0.38911876
153  0.020750671  0.188578879  0.33812314
154 -0.599976614  1.604588445 -1.18908484
155 -0.251498138  0.252942950  0.01086428
156  0.046160560 -1.742343256  0.11995057
157 -0.033699090  1.347132161 -1.26180903
158  0.953656591 -0.261969619 -0.38911876
159  1.795812908  1.089675876  0.77446827
161 -1.057354613 -0.905610331 -1.47998160
163 -1.434872962 -0.197605548  0.19267476
164 -0.690726217  0.317307022 -0.17094619
165  0.231289751  1.475860303 -0.17094619
166 -2.618247786  1.411496232 -0.38911876
167 -0.316837852 -0.390697761 -0.06185991
168 -0.316837852 -1.227430687  0.15631266
169 -0.102668789  0.252942950  0.73810618
170  1.414664575 -1.356158829 -1.00727437
172  1.781292971 -2.064163612  1.35626179
174 -0.436627328  0.639127378  0.81083037
175 -2.752557198  0.896583662 -1.62542998
176 -0.015549170 -0.648154046  0.73810618
177 -0.716136106  0.832219591  1.39262388
178  1.051666162  0.832219591  0.11995057
179  1.117005877 -1.356158829  0.08358847
180  0.071570449  1.218404018 -1.58906788
181 -0.959345042  0.767855520  1.57443436
183 -1.031944724  0.767855520 -0.31639457
```

185	0.859277004	0.446035164	-1.29817113
186	0.902836813	0.317307022	1.13808922
187	0.398269020	-0.326333690	-0.64365342
188	-0.563676772	-0.519425904	-1.33453322
189	0.176839989	-0.583789975	-1.15272275
190	1.643353574	-0.455061833	1.75624483
191	0.438198846	0.317307022	-1.15272275
192	-0.331357788	1.990772872	-0.97091227
194	2.068061717	0.896583662	0.04722638
196	-1.231593851	-0.519425904	0.88355456
197	-1.264263708	0.317307022	-1.47998160
198	-1.837801199	-0.068877405	-0.28003247
199	-1.438502946	-0.841246260	0.41084733
200	-0.422107392	-0.197605548	0.70174408
201	-0.734286026	1.218404018	0.84719246
202	-0.596346629	0.832219591	0.51993361
203	-1.264263708	-0.261969619	0.33812314
205	0.420048925	1.926408801	0.41084733
207	-0.948455089	2.376957300	-0.06185991
208	0.220399798	-0.905610331	1.31989969
209	0.071570449	-2.514712111	-0.53456714
210	-1.656301993	-1.742343256	-1.73451626
211	-1.169884121	-0.004513334	-1.00727437
212	0.655997893	-0.519425904	1.79260693
213	1.617943686	1.282768089	0.01086428
214	0.234919735	1.540224374	0.55629571
216	-1.903140914	-0.583789975	0.51993361
218	-1.177144089	-1.227430687	-0.89818808
219	0.768527401	0.832219591	-0.28003247
220	1.102485940	-0.583789975	0.37448523
221	1.091595988	0.381671093	1.86533111
222	-0.418477407	-0.068877405	-0.89818808
223	-0.033699090	0.703491449	1.82896902
224	-1.500212676	-0.969974402	-0.86182599
225	-1.264263708	0.574763306	0.48357152
227	0.078830417	-0.390697761	-1.29817113
229	-0.792365772	1.540224374	0.15631266
230	1.309395035	-1.742343256	1.28353760
231	-0.879485391	-0.648154046	-0.31639457
232	0.184099957	0.446035164	0.01086428
233	2.017241939	0.767855520	0.41084733
234	0.387379068	-0.133241477	0.26539895
235	0.670517829	-0.648154046	-0.24367038
236	-0.745175978	0.059850737	-1.33453322
238	0.739487528	1.668952517	-1.47998160
240	-1.340493375	-0.133241477	1.31989969
241	0.989956432	2.183865086	0.41084733
242	1.607053733	0.703491449	-0.20730828
243	-1.053724629	-0.068877405	-0.82546389
244	0.086090386	-0.068877405	-1.58906788
245	0.434568861	0.317307022	-1.40725741
246	0.877426924	-0.712518117	-0.93455018
247	-0.247868154	-0.326333690	-0.46184295
249	-0.806885709	-0.905610331	-0.60729133
251	0.572508258	0.767855520	1.31989969
252	-1.402203105	0.896583662	1.53807226
253	0.263959608	0.510399235	1.02900294
254	-0.705246153	-0.068877405	-1.62542998

```
255  0.724967591 -1.098702544  1.28353760
256  0.046160560 -0.133241477  0.44720942
257 -1.227963867  1.411496232 -1.04363646
258  0.394639036 -0.004513334 -0.24367038
260 -0.894005328 -0.133241477 -1.18908484
262 -0.679836264 -0.261969619 -0.97091227
263 -0.440257312 -0.326333690  0.30176104
264  0.067940465 -0.583789975 -1.55270579
265 -0.229718233  0.188578879 -1.44361950
266 -0.825035629  0.317307022 -1.22544694
267  0.550728353  0.188578879  0.88355456
268 -0.636276455  0.574763306  0.48357152
269 -0.382177566 -0.133241477 -0.42548085
271  1.599793765 -0.390697761 -1.55270579
273  1.959162193 -0.776882188 -1.29817113
274  0.891946861 -0.583789975  1.35626179
275 -0.131708662  0.639127378  0.88355456
276 -0.331357788 -1.163066616  1.82896902
277 -0.236978201  0.639127378  0.01086428
278  0.078830417  0.703491449 -0.75273970
279 -0.131708662 -0.712518117  1.61079645
280 -1.511102628  1.025311805 -0.42548085
282  1.309395035 -0.197605548  0.01086428
284 -0.806885709  0.639127378  1.50171017
285 -0.222458265 -1.227430687 -0.82546389
286  0.006230735  1.347132161 -1.55270579
287 -0.019179154 -0.519425904  1.79260693
288 -0.255128122 -1.935435470 -0.89818808
289 -0.218828281 -0.583789975 -1.04363646
290  0.423678909  1.154039947  0.30176104
291  0.692297734 -1.163066616  1.53807226
293  1.538084035 -0.776882188 -0.09822200
295  1.843002701  1.475860303  0.26539895
296 -1.224333883 -0.455061833 -1.22544694
297  0.227659766  0.124214808 -0.89818808
298 -1.638152073 -0.455061833  0.51993361
299  1.233165368  1.475860303 -0.20730828
300  0.659627877  0.639127378 -1.04363646
301  0.358339195 -0.583789975  0.33812314
302 -0.062738963 -1.677979185  0.88355456
304  0.881056909  0.510399235 -0.60729133
306  0.162320052 -0.648154046 -1.44361950
    Advertising
1    0.61892880
2    1.37360777
3    0.46799300
4   -0.43762177
5   -0.58855757
7   -1.04136496
9   -1.04136496
10  -1.04136496
11   0.31705720
12  -0.43762177
13  -0.73949336
14   0.61892880
15   0.61892880
16  -0.28668598
18   0.92080039
```

20	1.37360777
21	-0.73949336
22	0.76986459
23	-0.13575018
24	-1.04136496
25	1.37360777
26	-1.04136496
27	0.61892880
29	-1.04136496
31	-1.04136496
32	1.37360777
33	0.76986459
34	0.92080039
35	-1.04136496
36	0.61892880
37	-1.04136496
38	-0.28668598
40	-1.04136496
42	-1.04136496
43	-1.04136496
44	0.61892880
45	-0.13575018
46	-1.04136496
47	1.07173618
48	-1.04136496
49	-1.04136496
51	1.67547936
53	-0.58855757
54	0.92080039
55	0.92080039
56	-0.28668598
57	-1.04136496
58	-1.04136496
59	1.22267198
60	-0.43762177
62	-1.04136496
64	0.46799300
65	0.76986459
66	-1.04136496
67	-1.04136496
68	1.07173618
69	1.97735096
70	-1.04136496
71	1.22267198
73	-1.04136496
75	-0.28668598
76	2.43015834
77	0.46799300
78	0.76986459
79	-0.89042916
80	-1.04136496
81	1.37360777
82	-1.04136496
84	0.01518561
86	-1.04136496
87	0.31705720
88	0.01518561
89	0.01518561

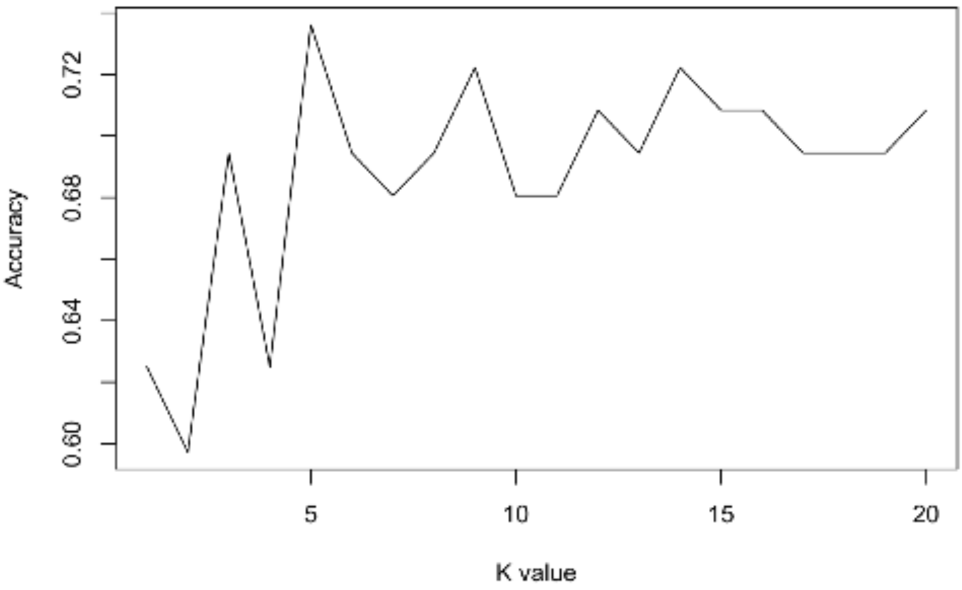
```
90  -0.58855757
91  -1.04136496
92   0.61892880
93  -1.04136496
95  -0.28668598
97   0.46799300
98  -0.28668598
99   2.58109414
100 -0.58855757
101  0.61892880
102 -1.04136496
103 -1.04136496
104 -1.04136496
106  0.16612141
108 -1.04136496
109 -0.73949336
110 -1.04136496
111  0.01518561
112  0.76986459
113 -0.28668598
114  0.61892880
115  0.31705720
117 -1.04136496
119 -0.73949336
120  0.16612141
121  0.61892880
122  0.46799300
123 -0.28668598
124 -1.04136496
125 -1.04136496
126 -1.04136496
128 -0.58855757
130  0.01518561
131  0.92080039
132 -0.58855757
133  0.31705720
134 -0.73949336
135 -1.04136496
136  1.07173618
137 -1.04136496
139  0.76986459
141  0.46799300
142 -1.04136496
143 -1.04136496
144  0.01518561
145 -1.04136496
146  0.61892880
147 -1.04136496
148  0.31705720
150  0.92080039
152  1.52454357
153 -1.04136496
154  0.01518561
155  0.46799300
156 -1.04136496
157 -1.04136496
158  0.16612141
159 -0.89042916
```

```
161 -1.04136496
163 -1.04136496
164 -1.04136496
165 -1.04136496
166  0.01518561
167  1.52454357
168 -1.04136496
169 -1.04136496
170  1.22267198
172  0.76986459
174 -0.28668598
175 -1.04136496
176 -1.04136496
177  0.31705720
178 -1.04136496
179  1.07173618
180 -0.58855757
181  1.22267198
183 -0.43762177
185  0.01518561
186  0.61892880
187 -1.04136496
188 -1.04136496
189 -1.04136496
190  1.67547936
191  0.92080039
192  0.92080039
194  0.01518561
196 -0.43762177
197 -0.13575018
198 -1.04136496
199 -0.28668598
200 -0.28668598
201 -1.04136496
202 -1.04136496
203 -0.43762177
205 -1.04136496
207 -1.04136496
208 -1.04136496
209 -1.04136496
210  0.61892880
211 -0.73949336
212  1.07173618
213  1.82641516
214 -0.28668598
216  1.22267198
218 -1.04136496
219  0.76986459
220  1.82641516
221  1.22267198
222 -1.04136496
223 -0.13575018
224  0.31705720
225 -1.04136496
227 -1.04136496
229  0.92080039
230 -1.04136496
231 -1.04136496
```

```
232 -1.04136496
233  0.46799300
234  1.67547936
235  0.61892880
236  0.16612141
238  0.16612141
240 -1.04136496
241 -1.04136496
242 -1.04136496
243 -1.04136496
244  0.92080039
245 -1.04136496
246 -1.04136496
247  1.97735096
249 -1.04136496
251  0.46799300
252 -0.28668598
253 -1.04136496
254 -0.28668598
255  2.43015834
256  0.16612141
257 -1.04136496
258  1.07173618
260  0.46799300
262 -0.43762177
263  1.22267198
264 -0.13575018
265 -0.28668598
266  0.46799300
267  0.76986459
268  0.01518561
269 -1.04136496
271 -1.04136496
273 -1.04136496
274  0.16612141
275 -0.73949336
276  0.61892880
277  1.07173618
278  0.76986459
279 -0.73949336
280  0.92080039
282  0.01518561
284 -1.04136496
285  0.61892880
286  0.61892880
287  0.61892880
288 -0.43762177
289 -1.04136496
290  2.73202993
291  1.07173618
293  1.37360777
295 -0.58855757
296  1.07173618
297  0.92080039
298  0.92080039
299 -1.04136496
300  1.52454357
301 -0.89042916
```

```
302 -1.04136496
304  1.37360777
306  2.88296573
 [ reached getOption("max.print") -- omitted 78 rows ]
attr(,"scaled:center")
     Sales   CompPrice      Income
  7.582835  125.070122   68.701220
Advertising
  6.899390
attr(,"scaled:scale")
     Sales   CompPrice      Income
  2.754833   15.536618   27.501166
Advertising
  6.625334
>
> classifier_knn <- knn(train = train_scale,
+                       test = test_scale,
+                       cl = train_cl$Urban,
+                       k = 1)
> classifier_knn
 [1] No  Yes Yes Yes No  No  Yes Yes Yes Yes
[11] Yes Yes Yes Yes Yes No  Yes Yes Yes No
[21] Yes No  Yes Yes Yes Yes No  Yes No  Yes
[31] Yes No  Yes Yes No  No  No  Yes Yes No
[41] No  No  No  Yes No  No  No  Yes Yes Yes
[51] Yes No  Yes Yes Yes No  Yes Yes Yes Yes
[61] Yes Yes Yes Yes Yes Yes Yes No  No  Yes
[71] Yes Yes
Levels: No Yes
>
> cm <- table(test_cl$Urban, classifier_knn)
> cm
     classifier_knn
      No Yes
  No   7   9
  Yes 16  40
>
> misClassError <- mean(classifier_knn != test_cl$Urban)
> print(paste('Accuracy =', 1-misClassError))
[1] "Accuracy = 0.652777777777778"
>
> accuracies <- vector()
>
> for(i in 1:20){
+   print(paste("For k = ",i))
+   classifier_knn <- knn(train = train_scale,
+                         test = test_scale,
+                         cl = train_cl$Urban,
+                         k = i)
+   misClassError <- mean(classifier_knn != test_cl$Urban)
+   print(paste('Accuracy =', 1-misClassError))
+   accuracies[i] <- 1-misClassError
+ }
[1] "For k =  1"
[1] "Accuracy = 0.652777777777778"
[1] "For k =  2"
[1] "Accuracy = 0.694444444444444"
[1] "For k =  3"
```

```
[1] "Accuracy = 0.75"
[1] "For k =  4"
[1] "Accuracy = 0.666666666666667"
[1] "For k =  5"
[1] "Accuracy = 0.708333333333333"
[1] "For k =  6"
[1] "Accuracy = 0.722222222222222"
[1] "For k =  7"
[1] "Accuracy = 0.708333333333333"
[1] "For k =  8"
[1] "Accuracy = 0.75"
[1] "For k =  9"
[1] "Accuracy = 0.722222222222222"
[1] "For k =  10"
[1] "Accuracy = 0.777777777777778"
[1] "For k =  11"
[1] "Accuracy = 0.791666666666667"
[1] "For k =  12"
[1] "Accuracy = 0.763888888888889"
[1] "For k =  13"
[1] "Accuracy = 0.791666666666667"
[1] "For k =  14"
[1] "Accuracy = 0.791666666666667"
[1] "For k =  15"
[1] "Accuracy = 0.791666666666667"
[1] "For k =  16"
[1] "Accuracy = 0.777777777777778"
[1] "For k =  17"
[1] "Accuracy = 0.777777777777778"
[1] "For k =  18"
[1] "Accuracy = 0.777777777777778"
[1] "For k =  19"
[1] "Accuracy = 0.777777777777778"
[1] "For k =  20"
[1] "Accuracy = 0.777777777777778"
> print(accuracies)
 [1] 0.6527778 0.6944444 0.7500000 0.6666667
 [5] 0.7083333 0.7222222 0.7083333 0.7500000
 [9] 0.7222222 0.7777778 0.7916667 0.7638889
[13] 0.7916667 0.7916667 0.7916667 0.7777778
[17] 0.7777778 0.7777778 0.7777778 0.7777778
> print(1:20)
 [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14
[15] 15 16 17 18 19 20
> plot(1:20,accuracies,ylab="Accuracy",xlab="K value", type='l')
```


Accuracy
0.60
0.64
0.68
0.72
5
10
15
20
K value

Lab Exercise No : 2
Exercise Title : Linear Regression
Date : 15-02-2021

Task:
To carry out Linear Regression analysis for the given Covid-19 Dataset

Program:

```
library(ISLR)

attach(covid_19_data)

plot(Confirmed~Recovered)

fit1=lm(Confirmed~Recovered)

abline(fit1)

summary(fit1)
fit2=lm(Confirmed~ Recovered + I(Recovered^2))
summary(fit2)
points(Recovered, fitted(fit2), col ="Red", pch=20)

fit3=lm(Confirmed~ Recovered + I(Recovered^2) + I(Recovered^3)) summary(fit3)
points(Deaths, fitted(fit3), col ="blue", pch=20)
fit4=lm(Confirmed~ poly(Recovered,4))
summary(fit4)

points(Recovered, fitted(fit4), col ="green", pch=20)
```

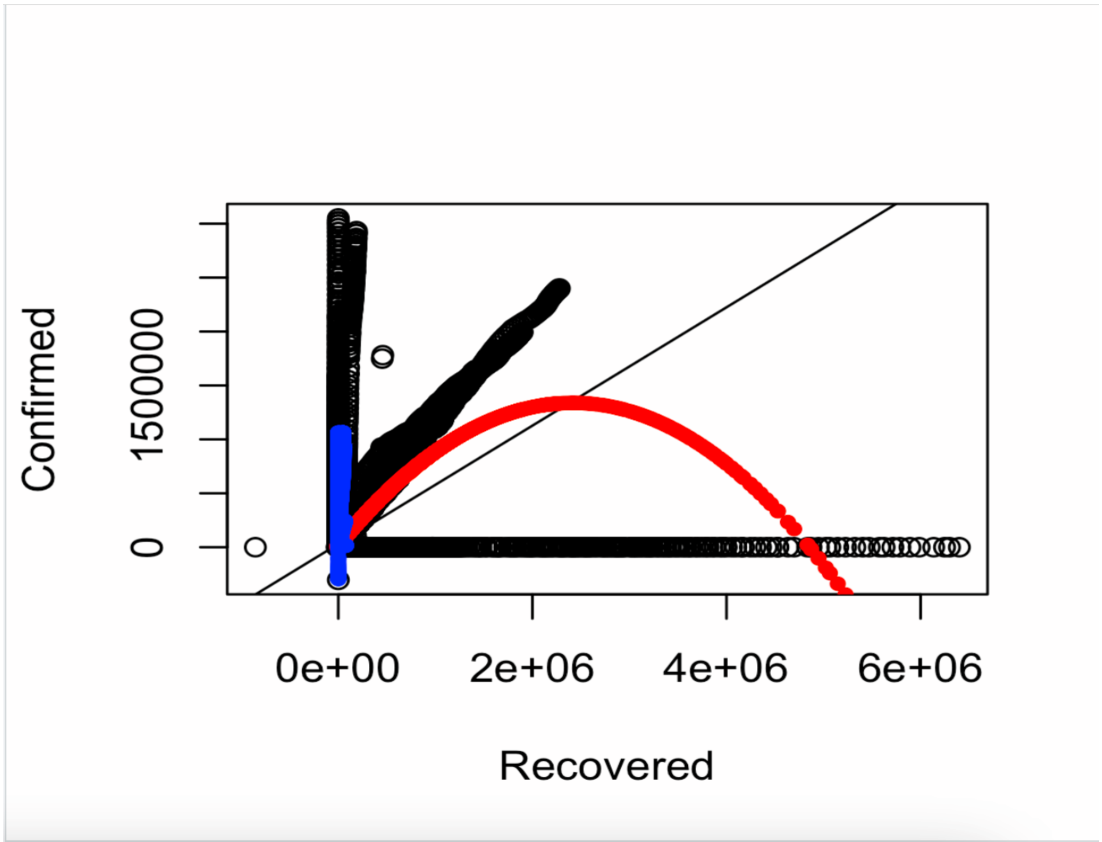
Output:

```
iew(covid_19_data)
> library(ISLR)
> attach(covid_19_data)
> plot(Confirmed~Recovered)
> fit1=lm(Confirmed~Recovered) > abline(fit1)
> summary(fit1)

Call:
lm(formula = Confirmed ~ Recovered)

Residuals:
Min 1Q Median 3Q Max

-3541331 -29857 -26719 -12076 3018813
```

```
Coefficients:
Estimate Std. Error t value Pr(>|t|)

(Intercept) 3.022e+04 2.783e+02 108.6 <2e-16 *** Recovered 5.487e-01 2.042e-03 268.7
<2e-16 *** ---
Signif. codes: 0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 123700 on 205949 degrees of freedom

Multiple R-squared: 0.2595, Adjusted R-squared: 0.2595 F-statistic: 7.219e+04 on 1 and
205949 DF, p-value: < 2.2e-16

> fit2=lm(Confirmed~ Recovered + I(Recovered^2)) > summary(fit2)

Call:
lm(formula = Confirmed ~ Recovered + I(Recovered^2))

Residuals:
Min 1Q Median 3Q Max

-1338450 -19409 -18642 -12489 3029618

Coefficients:
Estimate Std. Error t value Pr(>|t|)

(Intercept) 1.942e+04 2.510e+02 77.38 <2e-16 *** Recovered 1.091e+00 2.919e-03 373.58
<2e-16 *** I(Recovered^2) -2.254e-07 9.525e-10 -236.66 <2e-16 *** ---

Signif. codes: 0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 109700 on 205948 degrees of freedom Multiple R-squared: 0.4179,
Adjusted R-squared: 0.4178 F-statistic: 7.391e+04 on 2 and 205948 DF, p-value: < 2.2e-16

> points(Recovered, fitted(fit2), col ="Red", pch=20)

> fit3=lm(Confirmed~ Recovered + I(Recovered^2) + I(Recovered^3)) > summary(fit3)

Call:
lm(formula = Confirmed ~ Recovered + I(Recovered^2) + I(Recovered^3))

Residuals:
Min 1Q Median 3Q Max

-1062424 -16627 -16348 -13120 3032661

Coefficients:
Estimate Std. Error t value Pr(>|t|)

(Intercept) 1.638e+04 2.518e+02 65.04 <2e-16 *** Recovered 1.304e+00 4.194e-03 310.87
<2e-16 *** I(Recovered^2) -4.769e-07 3.711e-09 -128.51 <2e-16 *** I(Recovered^3)
4.186e-14 5.976e-16 70.05 <2e-16 *** ---

Signif. codes: 0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 108400 on 205947 degrees of freedom Multiple R-squared: 0.4314,
Adjusted R-squared: 0.4314 F-statistic: 5.208e+04 on 3 and 205947 DF, p-value: < 2.2e-16
```

```
> points(Deaths, fitted(fit3), col ="blue", pch=20) > fit4=lm(Confirmed~ poly(Recovered,4))
> summary(fit4)

Call:
lm(formula = Confirmed ~ poly(Recovered, 4))

Residuals:
Min 1Q Median 3Q Max

-1196983 -18103 -17694 -12773 3030938

Coefficients:
Estimate Std. Error t value Pr(>|t|)

(Intercept) 4.540e+04 2.379e+02 190.87 <2e-16 *** poly(Recovered, 4)1 3.323e+07
1.080e+05 307.81 <2e-16 *** poly(Recovered, 4)2 -2.595e+07 1.080e+05 -240.40 <2e-16 ***
poly(Recovered, 4)3 7.592e+06 1.080e+05 70.33 <2e-16 *** poly(Recovered, 4)4 4.349e+06
1.080e+05 40.29 <2e-16 *** ---

Signif. codes: 0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 108000 on 205946 degrees of freedom Multiple R-squared: 0.4358,
Adjusted R-squared: 0.4358 F-statistic: 3.978e+04 on 4 and 205946 DF, p-value: < 2.2e-16

> points(Recovered, fitted(fit4), col ="green", pch=20)
```

Plot:



Lab Exercise No : 3
Exercise Title : Binary Classification
Date : 22-02-2021

Task:

To analyze the given dataset using Logistic regression for binary classification and report the findings as a R markdown document.

Program:

```
library(ISLR)
dataset <- read.csv("/Users/tarunsidhu/Downloads/
Cleaned-Data.csv", header=TRUE)
dataset = dataset[, 1:20]

library(caTools)
set.seed(123)

split <- sample.split(dataset$Severity_Mild,
SplitRatio= 0.8)

training_set = subset(dataset, split=TRUE)
test_set = subset(dataset , split=FALSE)

classifier = glm(formula = Severity_Mild ~ .,
                 family = binomial,
                 data = training_set)

summary(classifier)

prob_pred = predict(classifier, type= 'response')

table(dataset$Severity_Mild, prob_pred > 0.5)
```

Output:

```
##
## Call:
## glm(formula = Severity_Mild ~ ., family =
binomial, data = training_set)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -0.7585  -0.7585  -0.7585  -0.1526   1.6651
```

```
##
## Coefficients: (2 not defined because of
singularities)
##                           Estimate Std. Error z
value Pr(>|z|)
## (Intercept)             -1.099e+00  1.605e-02
-68.45   <2e-16 ***
## Fever                   -2.566e-13  9.938e-03
0.00        1
## Tiredness                4.820e-14  9.938e-03
0.00        1
## Dry.Cough               -5.239e-14  9.938e-03
0.00        1
## Difficulty.in.Breathing  4.407e-15  9.938e-03
0.00        1
## Sore.Throat             -5.609e-14  9.938e-03
0.00        1
## None_Sympton            -6.175e-14  1.871e-02
0.00        1
## Pains                   -3.575e-14  9.428e-03
0.00        1
## Nasal.Congestion        -3.291e-14  9.428e-03
0.00        1
## Runny.Nose              -1.317e-14  9.428e-03
0.00        1
## Diarrhea                -3.936e-14  9.428e-03
0.00        1
## None_Experiencing       -4.065e-14  1.667e-02
0.00        1
## Age_0.9                 -1.000e-13  1.297e-02
0.00        1
## Age_10.19               -7.609e-14  1.297e-02
0.00        1
## Age_20.24               -1.274e-13  1.297e-02
0.00        1
## Age_25.59               -7.663e-14  1.297e-02
0.00        1
## Age_60.                         NA         NA
NA       NA
## Gender_Female           -2.476e-14  1.005e-02
0.00        1
## Gender_Male             -4.591e-14  1.005e-02
0.00        1
```

```
## Gender_Transgender              NA          NA
NA       NA
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05
'.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken
to be 1)
##
##     Null deviance: 356296  on 316799  degrees of
freedom
## Residual deviance: 356296  on 316782  degrees of
freedom
## AIC: 356332
##
## Number of Fisher Scoring iterations: 4
```

```
prob_pred = predict(classifier, type= 'response')

table(dataset$Severity_Mild, prob_pred > 0.5)
```

```
##
##        FALSE
##   0 237600
##   1  79200
```

Lab Exercise No : 4
Exercise Title : Multiclass Classification
Date : 01-03-2021

Task:
For the given dataset carry out analysis using Linear Discriminant Analysis and develop a model for multiclass classification. Present the ROC curve and AUC based performance metrics.

Program And Output:

```
library(pROC)
```

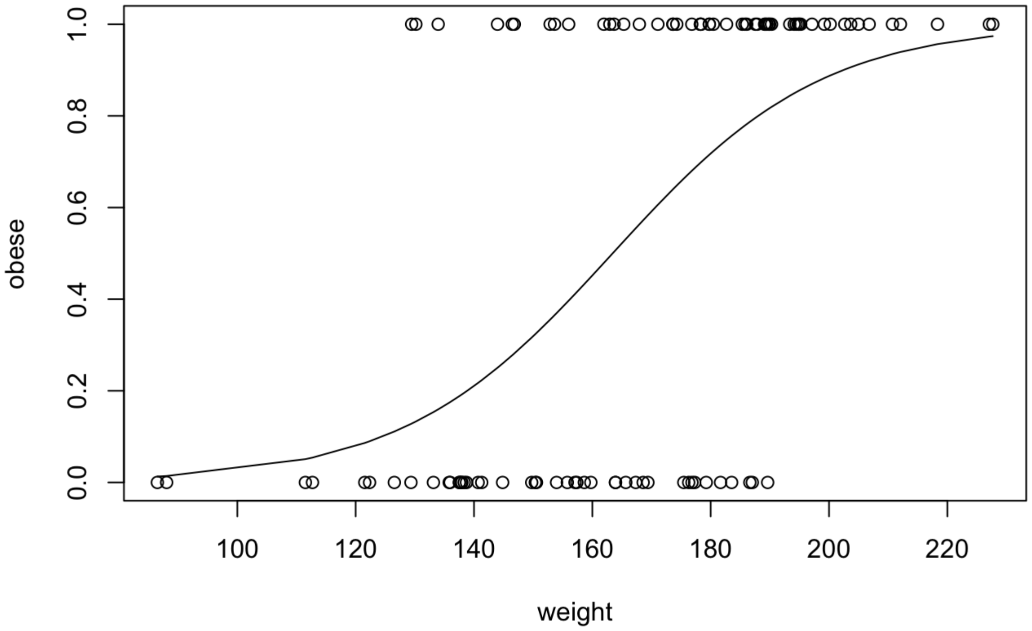
```
## Type 'citation("pROC")' for a citation.
##
## Attaching package: 'pROC'
## The following objects are masked from
'package:stats':
##
##     cov, smooth, var
```

```
library(randomForest)
```

```
## randomForest 4.6-14
```

```
## Type rfNews() to see new features/changes/bug
fixes.
set.seed(420)
num.samples <- 100
weight <- sort(rnorm(n=num.samples, mean=172,
sd=29))
obese <- ifelse(test=(runif(n=num.samples) <
(rank(weight)/num.samples)),
  yes=1, no=0)
obese
```

```
## ## ##

 [1] 0 0 0 0 0 0 0 0 1 1 0 1 0 0 0 0 0 0 0 0 0 1
0 1 1 0 0 0 1 1 0 0 1 0 0 0
[38] 0 1 1 1 0 0 1 0 0 1 0 0 1 1 1 1 0 0 1 0 0 1 1
1 0 1 1 1 0 1 0 1 1 1 0 0 1
[75] 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1
plot(x=weight, y=obese)
glm.fit=glm(obese ~ weight, family=binomial)
lines(weight, glm.fit$fitted.values)
```

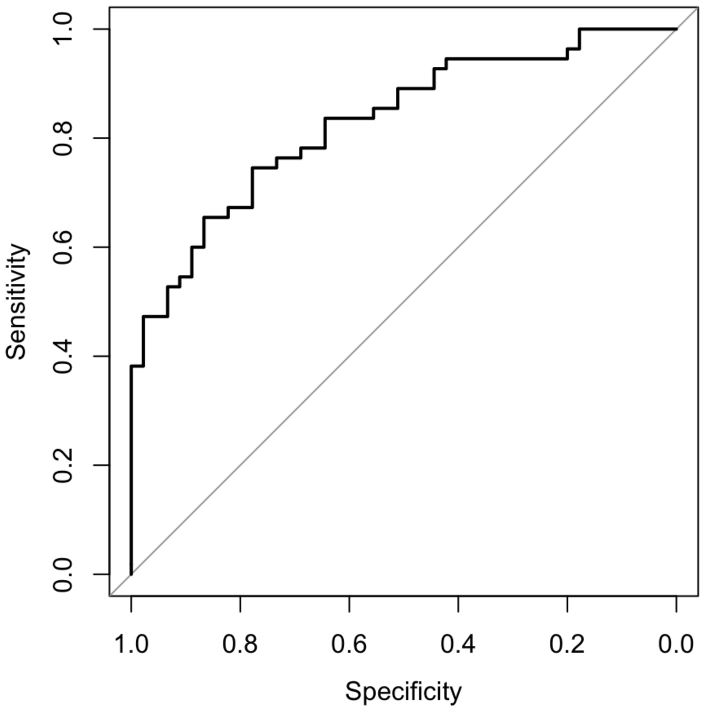


```
roc(obese, glm.fit$fitted.values, plot=TRUE)
```

```
## Setting levels: control = 0, case = 1
```

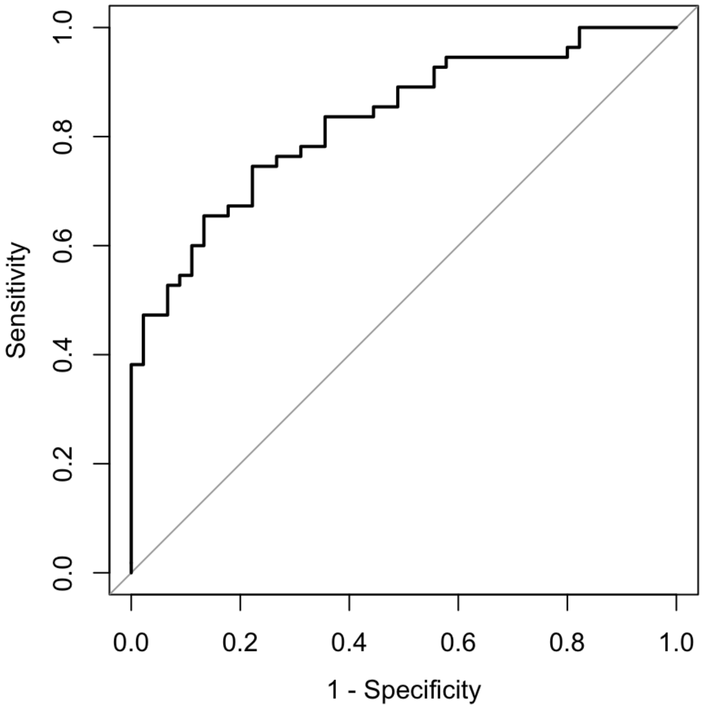
```
## Setting direction: controls < cases
##
```

```
## Call:
## roc.default(response = obese, predictor =
glm.fit$fitted.values,     plot = TRUE)
##
## Data: glm.fit$fitted.values in 45 controls
(obese 0) < 55 cases (obese 1).
## Area under the curve: 0.8291
par(pty = "s")
roc(obese, glm.fit$fitted.values, plot=TRUE)
## Setting levels: control = 0, case = 1
## Setting direction: controls < cases
```

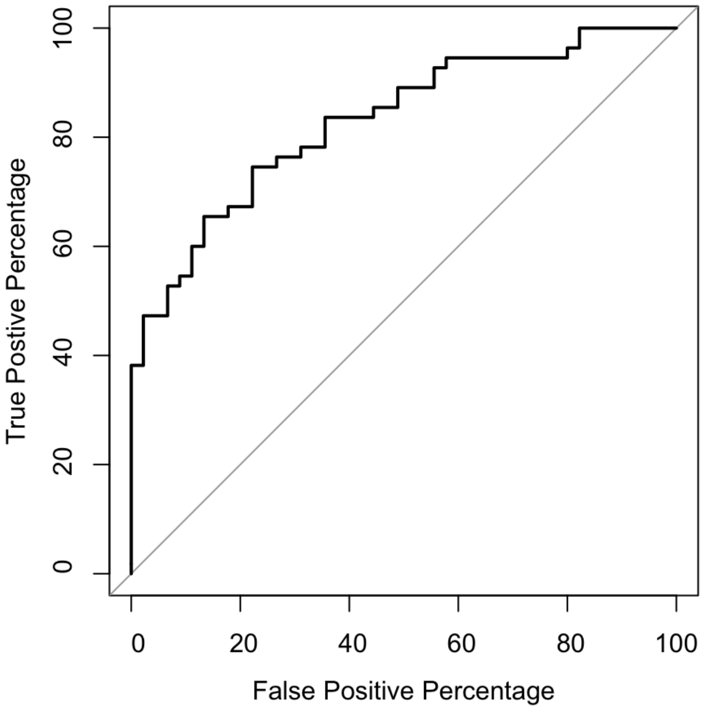


```
##
## Call:
## roc.default(response = obese, predictor =
glm.fit$fitted.values,     plot = TRUE)
##
## Data: glm.fit$fitted.values in 45 controls
(obese 0) < 55 cases (obese 1).
## Area under the curve: 0.8291
```

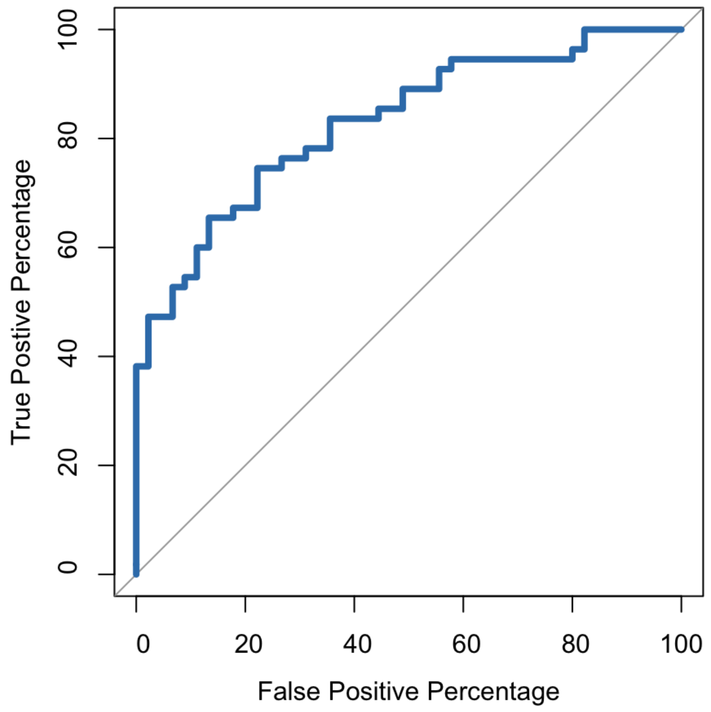
```
roc(obese, glm.fit$fitted.values, plot=TRUE,
legacy.axes=TRUE)
## Setting levels: control = 0, case = 1
## Setting direction: controls < cases
```



```
##
## Call:
## roc.default(response = obese, predictor =
glm.fit$fitted.values,     plot = TRUE,
legacy.axes = TRUE)
##
## Data: glm.fit$fitted.values in 45 controls
(obese 0) < 55 cases (obese 1).
## Area under the curve: 0.8291
roc(obese, glm.fit$fitted.values, plot=TRUE,
legacy.axes=TRUE, percent=TRUE, xlab="Fa
lse Positive Percentage", ylab="True Postive
Percentage")
## Setting levels: control = 0, case = 1
## Setting direction: controls < cases
```



```
##
## Call:
## roc.default(response = obese, predictor =
glm.fit$fitted.values,
E, plot = TRUE, legacy.axes = TRUE, xlab = "False
Positive Percentage",
rue Postive Percentage")
##
## Data: glm.fit$fitted.values in 45 controls
(obese 0) < 55 cases (obese 1).
## Area under the curve: 82.91%
percent = TRU
    ylab = "T
roc(obese, glm.fit$fitted.values, plot=TRUE,
legacy.axes=TRUE, percent=TRUE, xlab="Fa
lse Positive Percentage", ylab="True Postive
Percentage", col="#377eb8", lwd=4)
## Setting levels: control = 0, case = 1
## Setting direction: controls < cases
```



```
##
## Call:
## roc.default(response = obese, predictor =
glm.fit$fitted.values,
E, plot = TRUE, legacy.axes = TRUE, xlab = "False
Positive Percentage",
rue Postive Percentage", col = "#377eb8", lwd = 4)
##
## Data: glm.fit$fitted.values in 45 controls
(obese 0) < 55 cases (obese 1).
## Area under the curve: 82.91%
percent = TRU
    ylab = "T
```

```
roc.info <- roc(obese, glm.fit$fitted.values,
legacy.axes=TRUE)
## Setting levels: control = 0, case = 1
## Setting direction: controls < cases
## List of 15
##  $ percent
##  $ sensitivities
##  $ specificities
##  $ thresholds
##  $ direction
##  $ cases
##   ..- attr(*, "names")= chr [1:55] "9" "10"
"12" "23" ...
##  $ controls          : Named num [1:45]
0.0129 0.0141 0.0508 0.0542 0.0862 ...
##   ..- attr(*, "names")= chr [1:45] "1" "2"
"3" "4" ...
##  $ fun.sesp          :function (thresholds,
controls, cases, direction)
##  $ auc               : 'auc' num 0.829
 •      ##    ..- attr(*, "partial.auc")= logi
    FALSE
 •

 •      ##    ..- attr(*, "percent")= logi
    FALSE
 •

 •      ##    ..- attr(*, "roc")=List of 15
 •


: logi FALSE
: num [1:101] 1 1 1 1 1 ...
: num [1:101] 0 0.0222 0.0444 0.0667 0.0889 ...
: num [1:101] -Inf 0.0135 0.0325 0.0525
0.0702 ...
: chr "<"
: Named num [1:55] 0.128 0.133 0.159 0.25
0.278 ...
 •      ##    .. ..$ percent
 •

 •      ##    .. ..$ sensitivities
 •

 •      ##    .. ..$ specificities
 •

 •      ##    .. ..$ thresholds
```

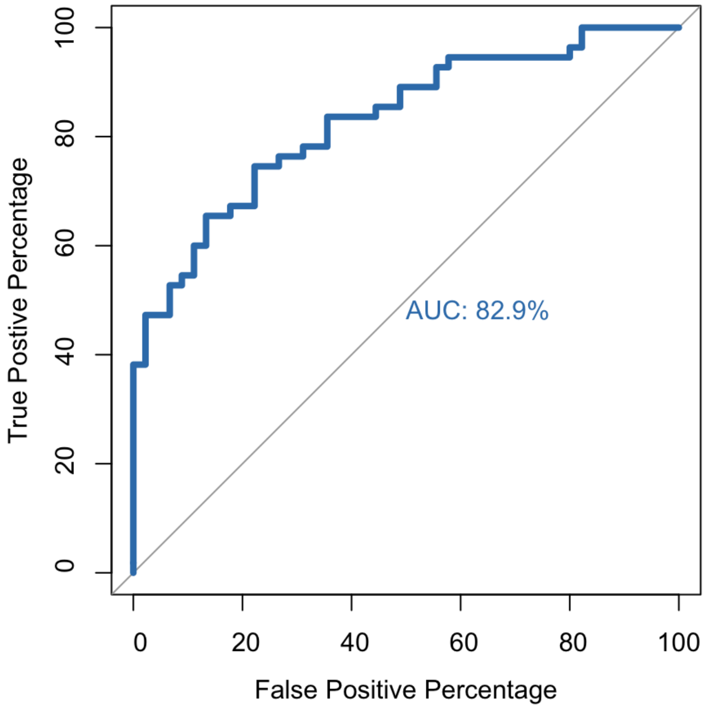
```
roc.df <- data.frame(
  tpp=roc.info$sensitivities*100,
  fpp=(1 - roc.info$specificities)*100,
  thresholds=roc.info$thresholds)
head(roc.df)
##   tpp       fpp thresholds
## 1 100 100.00000       -Inf
## 2 100  97.77778 0.01349011
## 3 100  95.55556 0.03245008
## 4 100  93.33333 0.05250145
## 5 100  91.11111 0.07017225
## 6 100  88.88889 0.08798755
```

```
tail(roc.df)
##          tpp fpp thresholds
## 96  9.090909   0  0.9275222
## 97  7.272727   0  0.9371857
## 98  5.454545   0  0.9480358
## 99  3.636364   0  0.9648800
## 100 1.818182   0  0.9735257
## 101 0.000000   0        Inf
```

```
roc.df[roc.df$tpp > 60 & roc.df$tpp < 80,]
##   ..- attr(*, "names")= chr [1:100] "1" "2" "3"
"4" ...
##  $ original.response : num [1:100] 0 0 0 0 0 0 0
0 1 1 ...
##  $ predictor          : Named num [1:100] 0.0129
0.0141 0.0508 0.0542 0.0862 ...
##   ..- attr(*, "names")= chr [1:100] "1" "2" "3"
"4" ...
##  $ response           : num [1:100] 0 0 0 0 0 0 0
0 1 1 ...
##  $ levels             : chr [1:2] "0" "1"
##  - attr(*, "class")= chr "roc"
```

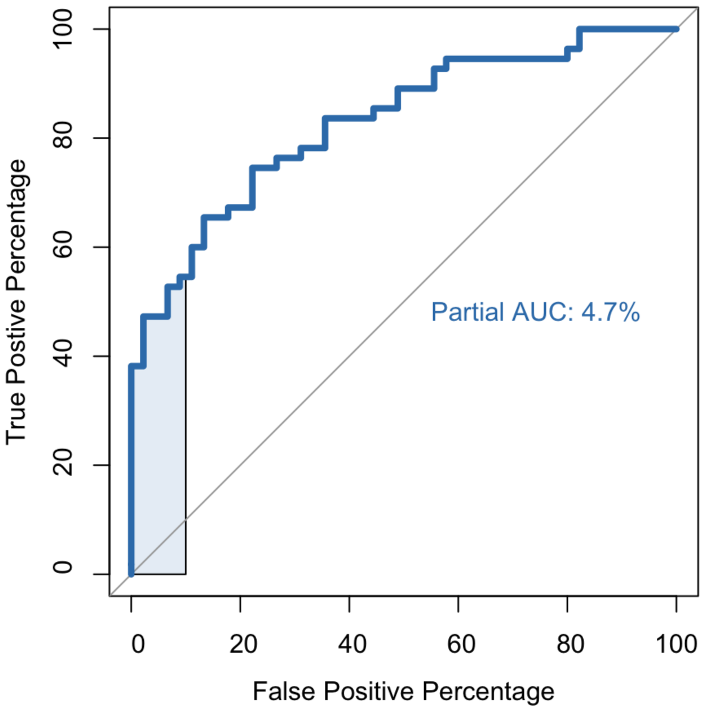
```
roc(obese, glm.fit$fitted.values, plot=TRUE,
legacy.axes=TRUE, percent=TRUE, xlab="Fa
lse Positive Percentage", ylab="True Postive
Percentage", col="#377eb8", lwd=4, prin
t.auc=TRUE)
```

```
## Setting levels: control = 0, case = 1
## Setting direction: controls < cases
```



```
##          tpp      fpp thresholds
## 42 78.18182 35.55556  0.5049310
## 43 78.18182 33.33333  0.5067116
## 44 78.18182 31.11111  0.5166680
## 45 76.36364 31.11111  0.5287933
## 46 76.36364 28.88889  0.5429351
## 47 76.36364 26.66667  0.5589494
## 48 74.54545 26.66667  0.5676342
## 49 74.54545 24.44444  0.5776086
## 50 74.54545 22.22222  0.5946054
## 51 72.72727 22.22222  0.6227449
## 52 70.90909 22.22222  0.6398136
## 53 69.09091 22.22222  0.6441654
## 54 67.27273 22.22222  0.6556705
## 55 67.27273 20.00000  0.6683618
## 56 67.27273 17.77778  0.6767661
## 57 65.45455 17.77778  0.6802060
## 58 65.45455 15.55556  0.6831936
## 59 65.45455 13.33333  0.6917225
## 60 63.63636 13.33333  0.6975300
## 61 61.81818 13.33333  0.6982807
```

```
roc(obese, glm.fit$fitted.values, plot=TRUE,
legacy.axes=TRUE, percent=TRUE, xlab="Fa
lse Positive Percentage", ylab="True Postive
Percentage", col="#377eb8", lwd=4, prin
t.auc=TRUE, print.auc.x=45, partial.auc=c(100, 90),
auc.polygon = TRUE, auc.polygon.c
ol = "#377eb822")
## Setting levels: control = 0, case = 1
## Setting direction: controls < cases
```

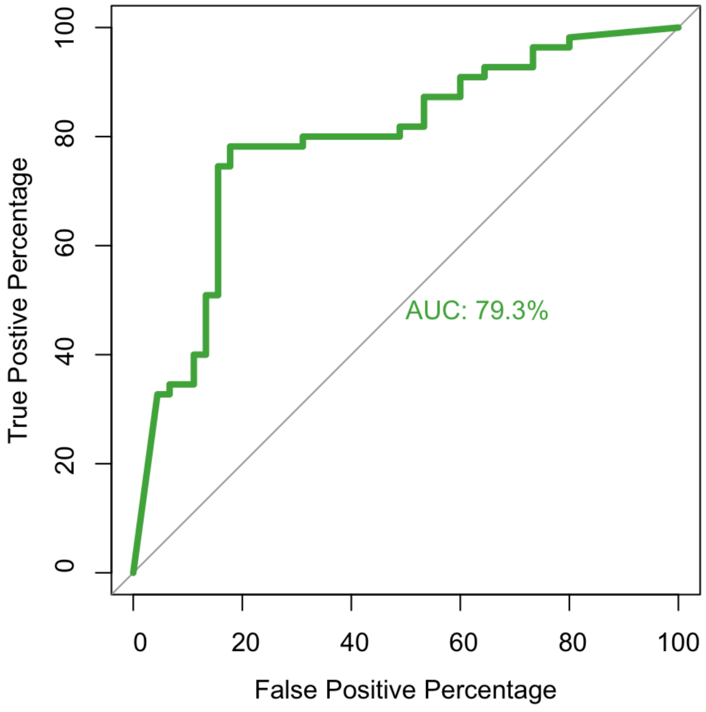


```
##
## Call:
## roc.default(response = obese, predictor =
glm.fit$fitted.values,     percent = TRU
E, plot = TRUE, legacy.axes = TRUE, xlab = "False
Positive Percentage",     ylab = "T
rue Postive Percentage", col = "#377eb8", lwd = 4,
print.auc = TRUE, print.auc.x
= 45, partial.auc = c(100,          90), auc.polygon
= TRUE, auc.polygon.col = "#377eb
822")
##
```

```
## Data: glm.fit$fitted.values in 45 controls
(obese 0) < 55 cases (obese 1).
## Partial area under the curve (specificity
100%-90%): 4.727%
##
## Call:
## roc.default(response = obese, predictor =
glm.fit$fitted.values,
E, plot = TRUE, legacy.axes = TRUE, xlab = "False
Positive Percentage",
rue Postive Percentage", col = "#377eb8", lwd = 4,
print.auc = TRUE)
##
## Data: glm.fit$fitted.values in 45 controls
(obese 0) < 55 cases (obese 1).
## Area under the curve: 82.91%
percent = TRU
    ylab = "T
```

```
## Setting levels: control = 0, case = 1
```

```
## Setting direction: controls > cases
```



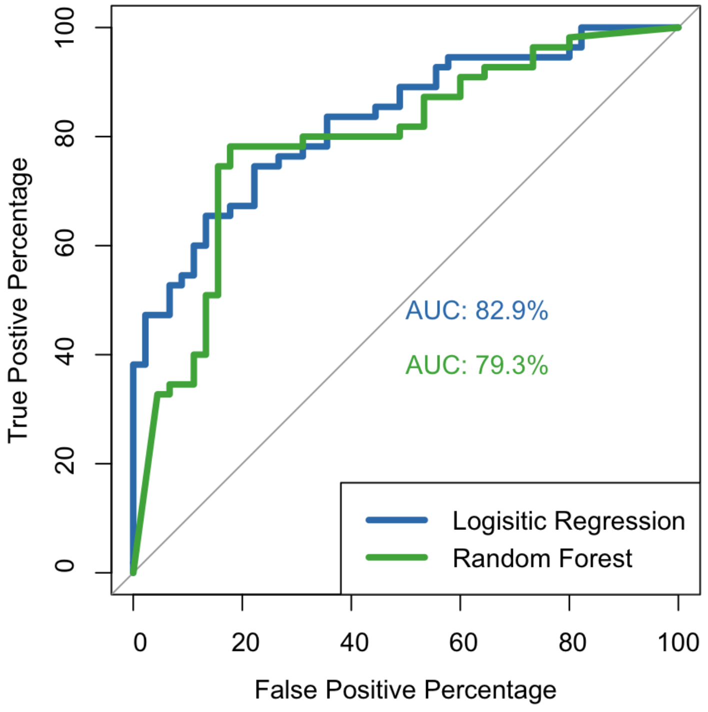
```
##
## Call:
## roc.default(response = obese, predictor =
rf.model$votes[, 1],
plot = TRUE, legacy.axes = TRUE, xlab = "False
Positive Percentage",
Postive Percentage", col = "#4daf4a", lwd = 4,
print.auc = TRUE)
##
## Data: rf.model$votes[, 1] in 45 controls (obese
0) > 55 cases (obese 1).
## Area under the curve: 79.29%
percent = TRUE,
   ylab = "True
roc(obese, glm.fit$fitted.values, plot=TRUE,
legacy.axes=TRUE, percent=TRUE, xlab="Fa
lse Positive Percentage", ylab="True Postive
Percentage", col="#377eb8", lwd=4, prin
t.auc=TRUE)
```

```
## Setting levels: control = 0, case = 1
rf.model <- randomForest(factor(obese) ~ weight)
roc(obese, rf.model$votes[,1], plot=TRUE,
legacy.axes=TRUE, percent=TRUE, xlab="False
Positive Percentage", ylab="True Postive
Percentage", col="#4daf4a", lwd=4, print.auc
=TRUE)
```

```
##
## Call:
## roc.default(response = obese, predictor =
glm.fit$fitted.values,
E, plot = TRUE, legacy.axes = TRUE, xlab = "False
Positive Percentage",
rue Postive Percentage", col = "#377eb8", lwd = 4,
print.auc = TRUE)
##
## Data: glm.fit$fitted.values in 45 controls
(obese 0) < 55 cases (obese 1).
## Area under the curve: 82.91%
percent = TRU
    ylab = "T
plot.roc(obese, rf.model$votes[,1], percent=TRUE,
col="#4daf4a", lwd=4, print.auc=TRU
E, add=TRUE, print.auc.y=40)
```

```
## Setting levels: control = 0, case = 1
```

```
## Setting direction: controls > cases
legend("bottomright", legend=c("Logisitic
Regression", "Random Forest"), col=c("#377e
b8", "#4daf4a"), lwd=4)
```


True Postive Percentage
0
20
40
60
80
100
False Positive Percentage
0
20
40
60
80
100
AUC: 82.9%
AUC: 79.3%
Logisitic Regression
Random Forest

Lab Exercise No : 5
Exercise Title : Model Selection & Resampling
Date : 15-03-2021

Task:

I) For the given dataset design binary classifiers that can predict the severity level of Covid-19. Demonstrate how resampling methods can improve the estimation of the model performance.

II) For the given dataset design binary classifiers that can predict the difficulty in breathing for Covid-19 cases. Demonstrate how model selection and regularisation techniques can be employed to improve the model accuracy.

Program And Output:

```
library(tidyverse)
library(boot)
Auto=read.csv("/Users/tarunsidhu/Desktop/Sem 4/ML/
ML(Lab)/Data Sets/Cleaned-Data.c sv")
names(Cleaned_Data)
ggplot(Cleaned_Data,
aes(Severity_Severe,Severity_None)) +

   geom_point() +
   geom_smooth(method = "lm", se = FALSE) +
   geom_smooth(method = "lm", formula = y ~ poly(x,
1), se = FALSE, linetype = 1) +
   geom_smooth(method = "lm", formula = y ~ poly(x,
2), se = FALSE, linetype = 2) +
   geom_smooth(method = "lm", formula = y ~ poly(x,
3), se = FALSE, linetype = 3)
 set.seed(1)
 sample <- sample(c(TRUE, FALSE),
nrow(Cleaned_Data), replace = T, prob = c(0.6,0.4
 ))
 train <- Cleaned_Data[sample, ]
 test <- Cleaned_Data[!sample, ]
 # loop for first ten polynomial
 mse.df <- tibble(degree = 1:10, mse = NA)
for(i in 1:10) {
lm.fit <- lm(Severity_None ~ poly(Severity_Severe,
i), data = train) mse.df[i, 2] <-
mean((test$Severity_None - predict(lm.fit,
test))^2)

}

 ggplot(mse.df, aes(degree, mse)) +
   geom_line() +
   geom_point() )
 glm.fit <- glm(Severity_None ~ Severity_Severe,
data = Cleaned_Data)
 coef(glm.fit)

glm.fit <- glm(Severity_None ~Severity_Severe ,
data = Cleaned_Data)
loocv.err <- cv.glm(Cleaned_Data, glm.fit)
str(loocv.err) loocv.err$delta[1] loocv_error <-
function(x) {

  glm.fit <- glm(Severity_None ~
poly(Severity_Severe, x), data =Cleaned_Data)
  cv.glm(auto, glm.fit)$delta[1]
}
library(purrr)
1:5 %>% map_dbl(loocv_error)

kfcv_error <- function(x) {
glm.fit <- glm(Severity_None ~
poly(Severity_Severe, x), data = Cleaned_Data)
cv.glm(auto, glm.fit, K = 10)$delta[1]

}
```

Lab Exercise No : 6
Exercise Title : Regularization for Regression models
Date : 22-03-2021

Task:

Apply Lasso and Ridge regression to the linear model generated for the given dataset

Program:

```
library(ISLR)
covid=read.csv("/Users/tarunsidhu/Desktop/Sem 4/ML/ML(Lab)/Data Sets/
covid_19_data.csv")
names(covid)
set.seed(1)
train=sample(392,196)
lm.fit=lm(SNo~Deaths,data=covid,subset=train)
attach(covid)
mean((SNo-predict(lm.fit,covid))[-train]^2)
lm.fit2=lm(SNo~poly(Deaths,2),data=covid,subset=train)
mean((SNo-predict(lm.fit2,covid))[-train]^2)
lm.fit3=lm(SNo~poly(Deaths,3),data=covid,subset=train)
mean((SNo-predict(lm.fit3,covid))[-train]^2)
set.seed(2)
train=sample(392,196)
lm.fit=lm(SNo~Deaths,subset=train)
mean((SNo-predict(lm.fit,covid))[-train]^2)
lm.fit2=lm(SNo~poly(Deaths,2),data=covid,subset=train)
mean((SNo-predict(lm.fit2,covid))[-train]^2)
lm.fit3=lm(SNo~poly(Deaths,3),data=covid,subset=train)
mean((SNo-predict(lm.fit3,covid))[-train]^2)
glm.fit=glm(SNo~Deaths,data=covid)
coef(glm.fit)
lm.fit=lm(SNo~Deaths,data=covid)
coef(lm.fit)
```

Output:

```
> library(ISLR)

Attaching package: ‘ISLR’

The following object is masked _by_ ‘.GlobalEnv’:

    Auto
```

```
> covid=read.csv("/Users/tarunsidhu/Desktop/Sem 4/ML/ML(Lab)/Data Sets/
covid_19_data.csv")
> names(covid)
[1] "SNo"             "ObservationDate" "Province.State"
[4] "Country.Region"  "Last.Update"     "Confirmed"
[7] "Deaths"          "Recovered"
> set.seed(1)
> train=sample(392,196)
> lm.fit=lm(SNo~Deaths,data=covid,subset=train)
> attach(covid)
> mean((SNo-predict(lm.fit,covid))[-train]^2)
[1] 13952167580
> lm.fit2=lm(SNo~poly(Deaths,2),data=covid,subset=train)
> mean((SNo-predict(lm.fit2,covid))[-train]^2)
[1] 1.156228e+13
> lm.fit3=lm(SNo~poly(Deaths,3),data=covid,subset=train)
> mean((SNo-predict(lm.fit3,covid))[-train]^2)
prediction from a rank-deficient fit may be misleading[1] 1.156228e+13
> set.seed(2)
> train=sample(392,196)
> lm.fit=lm(SNo~Deaths,subset=train)
> mean((SNo-predict(lm.fit,covid))[-train]^2)
[1] 13805978913
> lm.fit2=lm(SNo~poly(Deaths,2),data=covid,subset=train)
> mean((SNo-predict(lm.fit2,covid))[-train]^2)
[1] 3.382034e+12
> lm.fit3=lm(SNo~poly(Deaths,3),data=covid,subset=train)
> mean((SNo-predict(lm.fit3,covid))[-train]^2)
prediction from a rank-deficient fit may be misleading[1] 3.382034e+12
> glm.fit=glm(SNo~Deaths,data=covid)
> coef(glm.fit)
 (Intercept)       Deaths
1.000398e+05 2.325455e+00
> lm.fit=lm(SNo~Deaths,data=covid)
> coef(lm.fit)
 (Intercept)       Deaths
1.000398e+05 2.325455e+00
```

Lab Exercise No : 7
Exercise Title : Support Vector Machine
Date : 5-04-2021

Task:

Design a Support Vector Machine (for the given Dataset) - Binary Classifier using Linear and Radial Kernel and Cross validation to determine optimized values for the hyper-parameters of the model.

The task is to develop a binary classifier to predict each of the severity levels of i) mild ii) moderate iii) severe

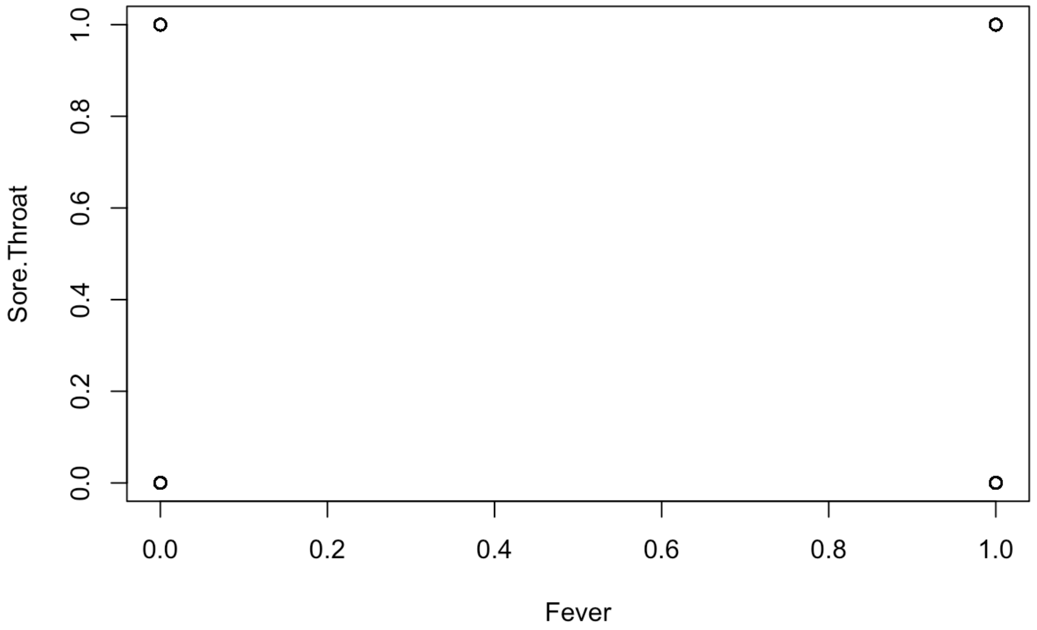
Program:
Output:

```
Cleaned.Data=read.csv("/Users/tarunsidhu/Desktop/Sem
4/ML/ML(Lab)/Data Sets/Cleaned-Data.csv")
summary(Cleaned.Data)
##      Fever          Tiredness      Dry.Cough
Difficulty.in.Breathing
##  Min.   :0.0000   Min.   :0.0   Min.   :0.0000
Min.   :0.0
##  1st Qu.:0.0000   1st Qu.:0.0   1st Qu.:0.0000
1st Qu.:0.0
##  Median :0.0000   Median :0.5   Median :1.0000
Median :0.5
##  Mean   :0.3125   Mean   :0.5   Mean   :0.5625
Mean   :0.5
##  3rd Qu.:1.0000   3rd Qu.:1.0   3rd Qu.:1.0000
3rd Qu.:1.0
##  Max.   :1.0000   Max.   :1.0   Max.   :1.0000
Max.   :1.0
##   Sore.Throat      None_Sympton        Pains
Nasal.Congestion
##  Min.   :0.0000   Min.   :0.0000   Min.   :0.0000
Min.   :0.0000
##  1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:0.0000
1st Qu.:0.0000
##  Median :0.0000   Median :0.0000   Median :0.0000
Median :1.0000
##  Mean   :0.3125   Mean   :0.0625   Mean   :0.3636
Mean   :0.5455
```

```
##  3rd Qu.:1.0000   3rd Qu.:0.0000   3rd Qu.:1.0000
3rd Qu.:1.0000
##  Max.   :1.0000   Max.   :1.0000   Max.   :1.0000
Max.   :1.0000
##    Runny.Nose        Diarrhea
None_Experiencing    Age_0.9
##  Min.   :0.0000   Min.   :0.0000   Min.
:0.00000   Min.   :0.0
##  1st Qu.:0.0000   1st Qu.:0.0000   1st
Qu.:0.00000   1st Qu.:0.0
##  Median :1.0000   Median :0.0000   Median
:0.00000   Median :0.0
##  Mean   :0.5455   Mean   :0.3636   Mean
:0.09091   Mean   :0.2
##  3rd Qu.:1.0000   3rd Qu.:1.0000   3rd
Qu.:0.00000   3rd Qu.:0.0
##  Max.   :1.0000   Max.   :1.0000   Max.
:1.00000   Max.   :1.0
##    Age_10.19     Age_20.24     Age_25.59
Age_60.    Gender_Female
##  Min.   :0.0   Min.   :0.0   Min.   :0.0   Min.
:0.0   Min.   :0.0000
##  1st Qu.:0.0   1st Qu.:0.0   1st Qu.:0.0   1st
Qu.:0.0   1st Qu.:0.0000
##  Median :0.0   Median :0.0   Median :0.0   Median
:0.0   Median :0.0000
##  Mean   :0.2   Mean   :0.2   Mean   :0.2   Mean
:0.2   Mean   :0.3333
##  3rd Qu.:0.0   3rd Qu.:0.0   3rd Qu.:0.0   3rd
Qu.:0.0   3rd Qu.:1.0000
##  Max.   :1.0   Max.   :1.0   Max.   :1.0   Max.
:1.0   Max.   :1.0000
##   Gender_Male     Gender_Transgender
Severity_Mild  Severity_Moderate
##  Min.   :0.0000   Min.   :0.0000     Min.   :0.00
Min.   :0.00
##  1st Qu.:0.0000   1st Qu.:0.0000     1st Qu.:0.00
1st Qu.:0.00
##  Median :0.0000   Median :0.0000     Median :0.00
Median :0.00
##  Mean   :0.3333   Mean   :0.3333     Mean   :0.25
Mean   :0.25
```

```
##   3rd Qu.:1.0000   3rd Qu.:1.0000     3rd Qu.:0.25
3rd Qu.:0.25
##   Max.   :1.0000   Max.   :1.0000     Max.   :1.00
Max.   :1.00
##   Severity_None  Severity_Severe Contact_Dont.Know
Contact_No
##   Min.   :0.00   Min.   :0.00     Min.   :0.0000
Min.   :0.0000
##   1st Qu.:0.00   1st Qu.:0.00     1st Qu.:0.0000
1st Qu.:0.0000
##   Median :0.00   Median :0.00     Median :0.0000
Median :0.0000
##   Mean   :0.25   Mean   :0.25     Mean   :0.3333
Mean   :0.3333
##   3rd Qu.:0.25   3rd Qu.:0.25     3rd Qu.:1.0000
3rd Qu.:1.0000
##   Max.   :1.00   Max.   :1.00     Max.   :1.0000
Max.   :1.0000
##    Contact_Yes       Country
##   Min.   :0.0000   Length:316800
##   1st Qu.:0.0000   Class :character
##   Median :0.0000   Mode  :character
##   Mean   :0.3333
##   3rd Qu.:1.0000
##   Max.   :1.0000
Cleaned_Data_1 <- Cleaned.Data[1:10000,]
attach(Cleaned_Data_1)
plot(Fever,Sore.Throat)
library(e1071)
svm.fit <- svm(Severity_Mild ~ Fever+Sore.Throat,
data = Cleaned_Data_1, type='C-classification',
kernel='linear', cost=10, scale=FALSE)
plot(svm.fit, Cleaned_Data_1)
## Error in plot.svm(svm.fit, Cleaned_Data_1):
missing formula.
summary(svm.fit)
##
## Call:
## svm(formula = Severity_Mild ~ Fever +
Sore.Throat, data = Cleaned_Data_1,
##     type = "C-classification", kernel = "linear",
cost = 10, scale = FALSE)
##
```

```
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  linear
##        cost:  10
##
## Number of Support Vectors:  5004
##
##  ( 2502 2502 )
##
##
## Number of Classes:  2
##
## Levels:
##  0 1
yhat <- predict(svm.fit, Cleaned_Data_1)
table(predict=yhat, truth=Severity_Mild)
##        truth
## predict    0    1
##       0 7498 2502
##       1    0    0
library(caret)
## Loading required package: lattice
## Loading required package: ggplot2
```



```
confusionMatrix(yhat,
as.factor(Cleaned_Data_1$Severity_Mild))
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 7498 2502
##          1    0    0
##
##                Accuracy : 0.7498
##                  95% CI : (0.7412, 0.7583)
##     No Information Rate : 0.7498
##     P-Value [Acc > NIR] : 0.5054
##
##                   Kappa : 0
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 1.0000
##             Specificity : 0.0000
##          Pos Pred Value : 0.7498
##          Neg Pred Value :    NaN
##              Prevalence : 0.7498
##          Detection Rate : 0.7498
##    Detection Prevalence : 1.0000
##       Balanced Accuracy : 0.5000
##
##        'Positive' Class : 0
##
```

Lab Exercise No : 8
Exercise Title : SVM vs Logistic Regression
Date : 12-04-2021

Task:

Compare the performance of the best SVM model from Ex7 with the performance of a Logistic Regression model for the same task. Present the performance metrics and ROC curves.

Program And Output:

```
#Logistic Regression for Cleaned.Data
library(tidyverse)
## ── Attaching packages
─────────────────────────────────────── tidyverse
1.3.0 ──
## ✓ ggplot2 3.3.3     ✓ purrr   0.3.4
## ✓ tibble  3.1.0     ✓ dplyr   1.0.5
## ✓ tidyr   1.1.3     ✓ stringr 1.4.0
## ✓ readr   1.4.0     ✓ forcats 0.5.1
## ── Conflicts
──────────────────────────────────────────
tidyverse_conflicts() ──
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
library(caret)
## Loading required package: lattice
##
## Attaching package: 'caret'
## The following object is masked from
'package:purrr':
##
##     lift
Cleaned.Data=read.csv("/Users/tarunsidhu/Desktop/Sem
4/ML/ML(Lab)/Data Sets/Cleaned-Data.csv")
Cleaned.Data <- na.omit(Cleaned.Data)
sample_n(Cleaned.Data, 3)
##   Fever Tiredness Dry.Cough
Difficulty.in.Breathing Sore.Throat None_Sympton
## 1     0         0         0
0            0            1
```

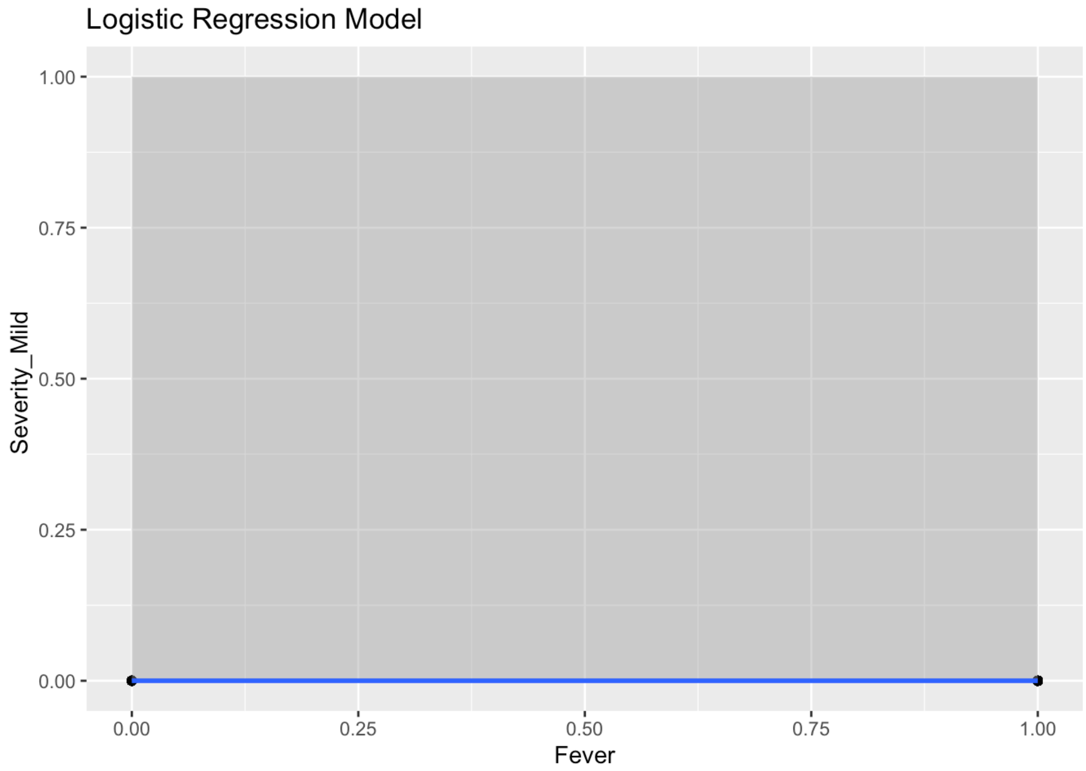
```
## 2     0         0         1
0           0            0
## 3     0         0         0
0           1            0
##   Pains Nasal.Congestion Runny.Nose Diarrhea
None_Experiencing Age_0.9
## 1     1                1          1        1
0       0
## 2     0                0          1        0
0       0
## 3     0                0          1        0
0       1
##   Age_10.19 Age_20.24 Age_25.59 Age_60.
Gender_Female Gender_Male
## 1         0         0         0       1
0           1
## 2         0         0         1       0
0           0
## 3         0         0         0       0
1           0
##   Gender_Transgender Severity_Mild
Severity_Moderate Severity_None
## 1                  0             1
0             0
## 2                  1             0
0             1
## 3                  0             1
0             0
##   Severity_Severe Contact_Dont.Know Contact_No
Contact_Yes Country
## 1               0                 0          0
1     UAE
## 2               0                 1          0
0 Germany
## 3               0                 0          1
0   Spain
```

```
set.seed(123)
training.samples <- Cleaned.Data$Severity_Mild %>%
  createDataPartition(p = 0.8, list = FALSE)
train.data  <- Cleaned.Data[training.samples, ]
test.data <- Cleaned.Data[-training.samples, ]
model <- glm( Severity_Mild ~., data = train.data,
family = binomial)
```

```
## Warning: glm.fit: algorithm did not converge
summary(model)
##
## Call:
## glm(formula = Severity_Mild ~ ., family =
binomial, data = train.data)
##
## Deviance Residuals:
##        Min          1Q      Median          3Q
Max
## -2.409e-06  -2.409e-06  -2.409e-06  -1.204e-06
2.409e-06
##
## Coefficients: (3 not defined because of
singularities)
##                              Estimate Std. Error z
value Pr(>|z|)
## (Intercept)                 2.657e+01  3.831e+03
0.007    0.994
## Fever                      -2.835e-11  1.712e+03
0.000    1.000
## Tiredness                  -9.100e-12  1.713e+03
0.000    1.000
## Dry.Cough                  -1.165e-11  1.714e+03
0.000    1.000
## Difficulty.in.Breathing    -8.587e-12  1.714e+03
0.000    1.000
## Sore.Throat                -3.166e-11  1.714e+03
0.000    1.000
## None_Sympton               -3.347e-11  3.225e+03
0.000    1.000
## Pains                      -2.857e-11  1.626e+03
0.000    1.000
## Nasal.Congestion           -9.934e-12  1.626e+03
0.000    1.000
## Runny.Nose                 -8.456e-12  1.626e+03
0.000    1.000
## Diarrhea                   -2.654e-11  1.626e+03
0.000    1.000
## None_Experiencing          -2.350e-11  2.872e+03
0.000    1.000
## Age_0.9                    -4.113e-11  2.238e+03
0.000    1.000
```

```
## Age_10.19                 -1.772e-12  2.237e+03
0.000    1.000
## Age_20.24                 -8.467e-13  2.239e+03
0.000    1.000
## Age_25.59                 -5.599e-12  2.237e+03
0.000    1.000
## Age_60.                           NA         NA
NA       NA
## Gender_Female              3.834e-13  1.732e+03
0.000    1.000
## Gender_Male               -2.137e-11  1.734e+03
0.000    1.000
## Gender_Transgender                NA         NA
NA       NA
## Severity_Moderate         -5.313e+01  2.001e+03
-0.027    0.979
## Severity_None             -5.313e+01  2.001e+03
-0.027    0.979
## Severity_Severe           -5.313e+01  2.000e+03
-0.027    0.979
## Contact_Dont.Know          2.310e-11  1.733e+03
0.000    1.000
## Contact_No                 2.310e-11  1.733e+03
0.000    1.000
## Contact_Yes                       NA         NA
NA       NA
## CountryFrance              7.232e-11  3.165e+03
0.000    1.000
## CountryGermany             7.161e-11  3.163e+03
0.000    1.000
## CountryIran                7.133e-11  3.166e+03
0.000    1.000
## CountryItaly               7.202e-11  3.156e+03
0.000    1.000
## CountryOther               7.180e-11  3.160e+03
0.000    1.000
## CountryOther-EUR           7.138e-11  3.165e+03
0.000    1.000
## CountryRepublic of Korean  7.096e-11  3.163e+03
0.000    1.000
## CountrySpain               7.149e-11  3.160e+03
0.000    1.000
```

```
## CountryUAE                  7.131e-11  3.164e+03
0.000    1.000
##
## (Dispersion parameter for binomial family taken
to be 1)
##
##     Null deviance: 2.8504e+05  on 253439  degrees
of freedom
## Residual deviance: 1.4704e-06  on 253408  degrees
of freedom
## AIC: 64
##
## Number of Fisher Scoring iterations: 25
probabilities <- model %>% predict(test.data, type =
"response")
## Warning in predict.lm(object, newdata, se.fit,
scale = 1, type = if (type == :
## prediction from a rank-deficient fit may be
misleading
predicted.classes <- ifelse(probabilities > 0.5,
"pos", "neg")
mean(predicted.classes == test.data$Severity_Mild)
## [1] 0
model <- glm( Severity_Mild ~ Fever, data =
train.data, family = binomial)
summary(model)$coef
##                 Estimate  Std. Error      z value
Pr(>|z|)
## (Intercept) -1.100228807 0.005536898 -198.7085344
0.0000000
## Fever        0.005159614 0.009887642    0.5218245
0.6017925
newdata <- data.frame(Fever = c(20,  180))
probabilities <- model %>% predict(newdata, type =
"response")
predicted.classes <- ifelse(probabilities > 0.5,
"pos", "neg")
predicted.classes
##     1     2
## "neg" "neg"
train.data %>%
  mutate(prob = ifelse(Severity_Mild == "pos", 1,
0)) %>%
  ggplot(aes(Fever, prob)) +
  geom_point(alpha = 0.2) +
  geom_smooth(method = "glm", method.args =
list(family = "binomial")) +
  labs(
    title = "Logistic Regression Model",
    x = "Fever",
    y = "Severity_Mild "
  )
## `geom_smooth()` using formula 'y ~ x'
## Warning: glm.fit: algorithm did not converge
```



```
model <- glm( Severity_Mild ~ Fever+Sore.Throat,
              data = train.data, family = binomial)
summary(model)$coef
##                 Estimate  Std. Error      z value
Pr(>|z|)
## (Intercept) -1.098917084 0.006628773 -165.7798718
0.0000000
## Fever        0.004565438 0.010024571    0.4554248
0.6488037
## Sore.Throat -0.003613004 0.010044562   -0.3596975
0.7190733
model <- glm( Severity_Mild ~., data = train.data,
family = binomial)
## Warning: glm.fit: algorithm did not converge
```

```
summary(model)$coef
##                                 Estimate Std.
Error       z value  Pr(>|z|)
## (Intercept)                 2.656606e+01
3831.079  6.934355e-03 0.9944672
## Fever                      -2.834899e-11
1712.241 -1.655666e-14 1.0000000
## Tiredness                  -9.099505e-12
1713.002 -5.312022e-15 1.0000000
## Dry.Cough                  -1.165313e-11
1714.381 -6.797282e-15 1.0000000
## Difficulty.in.Breathing   -8.586718e-12
1714.068 -5.009556e-15 1.0000000
## Sore.Throat                -3.165622e-11
1713.610 -1.847341e-14 1.0000000
## None_Sympton               -3.347069e-11
3224.916 -1.037878e-14 1.0000000
## Pains                      -2.856694e-11
1626.117 -1.756759e-14 1.0000000
## Nasal.Congestion           -9.933926e-12
1625.700 -6.110552e-15 1.0000000
## Runny.Nose                 -8.456039e-12
1625.676 -5.201552e-15 1.0000000
## Diarrhea                   -2.653532e-11
1626.142 -1.631796e-14 1.0000000
## None_Experiencing          -2.349686e-11
2871.525 -8.182709e-15 1.0000000
## Age_0.9                    -4.112961e-11
2238.427 -1.837434e-14 1.0000000
## Age_10.19                  -1.772394e-12
2236.878 -7.923515e-16 1.0000000
## Age_20.24                  -8.466565e-13
2238.713 -3.781890e-16 1.0000000
## Age_25.59                  -5.598877e-12
2236.504 -2.503406e-15 1.0000000
## Gender_Female               3.834451e-13
1732.163  2.213678e-16 1.0000000
## Gender_Male                -2.137028e-11
1733.744 -1.232609e-14 1.0000000
## Severity_Moderate          -5.313213e+01
2001.004 -2.655274e-02 0.9788165
## Severity_None              -5.313213e+01
2001.113 -2.655129e-02 0.9788176
```

```
## Severity_Severe           -5.313213e+01
2000.442 -2.656019e-02 0.9788105
## Contact_Dont.Know          2.310094e-11
1733.461  1.332648e-14 1.0000000
## Contact_No                 2.310135e-11
1732.623  1.333317e-14 1.0000000
## CountryFrance              7.231626e-11
3164.549  2.285200e-14 1.0000000
## CountryGermany             7.160902e-11
3163.358  2.263703e-14 1.0000000
## CountryIran                7.133118e-11
3165.502  2.253392e-14 1.0000000
## CountryItaly               7.202028e-11
3156.092  2.281945e-14 1.0000000
## CountryOther               7.179578e-11
3159.935  2.272065e-14 1.0000000
## CountryOther-EUR           7.137971e-11
3164.704  2.255494e-14 1.0000000
## CountryRepublic of Korean  7.095567e-11
3162.669  2.243537e-14 1.0000000
## CountrySpain               7.148842e-11
3159.955  2.262324e-14 1.0000000
## CountryUAE                 7.131168e-11
3164.062  2.253802e-14 1.0000000
coef(model)
##               (Intercept)
Fever                 Tiredness
##              2.656606e+01
-2.834899e-11             -9.099505e-12
##                 Dry.Cough
Difficulty.in.Breathing               Sore.Throat
##             -1.165313e-11
-8.586718e-12             -3.165622e-11
##              None_Sympton
Pains          Nasal.Congestion
##             -3.347069e-11
-2.856694e-11             -9.933926e-12
##                Runny.Nose
Diarrhea         None_Experiencing
##             -8.456039e-12
-2.653532e-11             -2.349686e-11
##                   Age_0.9
Age_10.19                 Age_20.24
```

```
##             -4.112961e-11
-1.772394e-12             -8.466565e-13
##                 Age_25.59
Age_60.             Gender_Female
##             -5.598877e-12
NA              3.834451e-13
##               Gender_Male
Gender_Transgender         Severity_Moderate
##             -2.137028e-11
NA             -5.313213e+01
##             Severity_None
Severity_Severe         Contact_Dont.Know
##             -5.313213e+01
-5.313213e+01              2.310094e-11
##                Contact_No
Contact_Yes             CountryFrance
##              2.310135e-11
NA              7.231626e-11
##            CountryGermany
CountryIran              CountryItaly
##              7.160902e-11
7.133118e-11              7.202028e-11
##              CountryOther          CountryOther-
EUR CountryRepublic of Korean
##              7.179578e-11
7.137971e-11              7.095567e-11
##              CountrySpain
CountryUAE
##              7.148842e-11
7.131168e-11
summary(model )$coef
##                                Estimate Std.
Error       z value  Pr(>|z|)
## (Intercept)                2.656606e+01
3831.079  6.934355e-03 0.9944672
## Fever                     -2.834899e-11
1712.241 -1.655666e-14 1.0000000
## Tiredness                 -9.099505e-12
1713.002 -5.312022e-15 1.0000000
## Dry.Cough                 -1.165313e-11
1714.381 -6.797282e-15 1.0000000
## Difficulty.in.Breathing   -8.586718e-12
1714.068 -5.009556e-15 1.0000000
```

```
## Sore.Throat                  -3.165622e-11
1713.610 -1.847341e-14 1.0000000
## None_Sympton                 -3.347069e-11
3224.916 -1.037878e-14 1.0000000
## Pains                        -2.856694e-11
1626.117 -1.756759e-14 1.0000000
## Nasal.Congestion             -9.933926e-12
1625.700 -6.110552e-15 1.0000000
## Runny.Nose                   -8.456039e-12
1625.676 -5.201552e-15 1.0000000
## Diarrhea                     -2.653532e-11
1626.142 -1.631796e-14 1.0000000
## None_Experiencing            -2.349686e-11
2871.525 -8.182709e-15 1.0000000
## Age_0.9                      -4.112961e-11
2238.427 -1.837434e-14 1.0000000
## Age_10.19                    -1.772394e-12
2236.878 -7.923515e-16 1.0000000
## Age_20.24                    -8.466565e-13
2238.713 -3.781890e-16 1.0000000
## Age_25.59                    -5.598877e-12
2236.504 -2.503406e-15 1.0000000
## Gender_Female                 3.834451e-13
1732.163  2.213678e-16 1.0000000
## Gender_Male                  -2.137028e-11
1733.744 -1.232609e-14 1.0000000
## Severity_Moderate            -5.313213e+01
2001.004 -2.655274e-02 0.9788165
## Severity_None                -5.313213e+01
2001.113 -2.655129e-02 0.9788176
## Severity_Severe              -5.313213e+01
2000.442 -2.656019e-02 0.9788105
## Contact_Dont.Know             2.310094e-11
1733.461  1.332648e-14 1.0000000
## Contact_No                    2.310135e-11
1732.623  1.333317e-14 1.0000000
## CountryFrance                 7.231626e-11
3164.549  2.285200e-14 1.0000000
## CountryGermany                7.160902e-11
3163.358  2.263703e-14 1.0000000
## CountryIran                   7.133118e-11
3165.502  2.253392e-14 1.0000000
```

```
## CountryItaly                   7.202028e-11
3156.092  2.281945e-14 1.0000000
## CountryOther                   7.179578e-11
3159.935  2.272065e-14 1.0000000
## CountryOther-EUR               7.137971e-11
3164.704  2.255494e-14 1.0000000
## CountryRepublic of Korean  7.095567e-11
3162.669  2.243537e-14 1.0000000
## CountrySpain                   7.148842e-11
3159.955  2.262324e-14 1.0000000
## CountryUAE                     7.131168e-11
3164.062  2.253802e-14 1.0000000
```

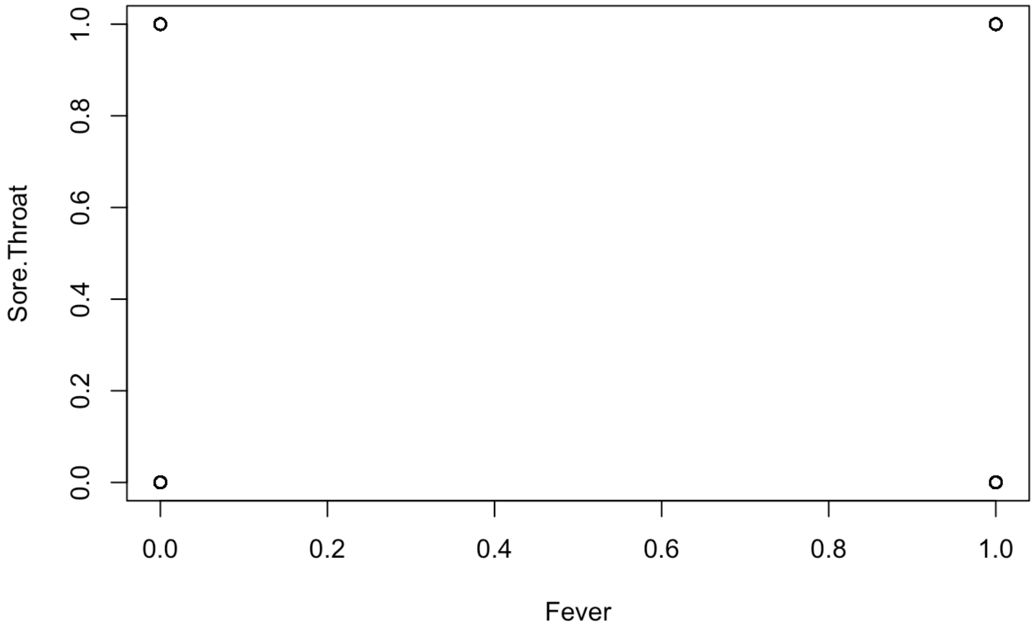
```
#SVM for Cleaned.Data
Cleaned.Data=read.csv("/Users/tarunsidhu/Desktop/Sem
4/ML/ML(Lab)/Data Sets/Cleaned-Data.csv")
summary(Cleaned.Data)
```

```
##      Fever          Tiredness     Dry.Cough
Difficulty.in.Breathing
##  Min.   :0.0000   Min.   :0.0   Min.   :0.0000
Min.   :0.0
##  1st Qu.:0.0000   1st Qu.:0.0   1st Qu.:0.0000
1st Qu.:0.0
##  Median :0.0000   Median :0.5   Median :1.0000
Median :0.5
##  Mean   :0.3125   Mean   :0.5   Mean   :0.5625
Mean   :0.5
##  3rd Qu.:1.0000   3rd Qu.:1.0   3rd Qu.:1.0000
3rd Qu.:1.0
##  Max.   :1.0000   Max.   :1.0   Max.   :1.0000
Max.   :1.0
##   Sore.Throat      None_Sympton         Pains
Nasal.Congestion
##  Min.   :0.0000   Min.   :0.0000   Min.   :0.0000
Min.   :0.0000
##  1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:0.0000
1st Qu.:0.0000
##  Median :0.0000   Median :0.0000   Median :0.0000
Median :1.0000
##  Mean   :0.3125   Mean   :0.0625   Mean   :0.3636
Mean   :0.5455
##  3rd Qu.:1.0000   3rd Qu.:0.0000   3rd Qu.:1.0000
3rd Qu.:1.0000
```

```
##  Max.   :1.0000   Max.   :1.0000   Max.   :1.0000
Max.   :1.0000
##    Runny.Nose        Diarrhea
None_Experiencing    Age_0.9
##  Min.   :0.0000   Min.   :0.0000   Min.
:0.00000   Min.   :0.0
##  1st Qu.:0.0000   1st Qu.:0.0000   1st
Qu.:0.00000   1st Qu.:0.0
##  Median :1.0000   Median :0.0000   Median
:0.00000   Median :0.0
##  Mean   :0.5455   Mean   :0.3636   Mean
:0.09091   Mean   :0.2
##  3rd Qu.:1.0000   3rd Qu.:1.0000   3rd
Qu.:0.00000   3rd Qu.:0.0
##  Max.   :1.0000   Max.   :1.0000   Max.
:1.00000   Max.   :1.0
##    Age_10.19     Age_20.24     Age_25.59
Age_60.    Gender_Female
##  Min.   :0.0   Min.   :0.0   Min.   :0.0   Min.
:0.0   Min.   :0.0000
##  1st Qu.:0.0   1st Qu.:0.0   1st Qu.:0.0   1st
Qu.:0.0   1st Qu.:0.0000
##  Median :0.0   Median :0.0   Median :0.0   Median
:0.0   Median :0.0000
##  Mean   :0.2   Mean   :0.2   Mean   :0.2   Mean
:0.2   Mean   :0.3333
##  3rd Qu.:0.0   3rd Qu.:0.0   3rd Qu.:0.0   3rd
Qu.:0.0   3rd Qu.:1.0000
##  Max.   :1.0   Max.   :1.0   Max.   :1.0   Max.
:1.0   Max.   :1.0000
##   Gender_Male     Gender_Transgender
Severity_Mild  Severity_Moderate
##  Min.   :0.0000   Min.   :0.0000     Min.   :0.00
Min.   :0.00
##  1st Qu.:0.0000   1st Qu.:0.0000     1st Qu.:0.00
1st Qu.:0.00
##  Median :0.0000   Median :0.0000     Median :0.00
Median :0.00
##  Mean   :0.3333   Mean   :0.3333     Mean   :0.25
Mean   :0.25
##  3rd Qu.:1.0000   3rd Qu.:1.0000     3rd Qu.:0.25
3rd Qu.:0.25
```

```
##  Max.   :1.0000   Max.   :1.0000    Max.   :1.00
Max.   :1.00
##  Severity_None   Severity_Severe Contact_Dont.Know
Contact_No
##  Min.   :0.00   Min.   :0.00    Min.   :0.0000
Min.   :0.0000
##  1st Qu.:0.00   1st Qu.:0.00    1st Qu.:0.0000
1st Qu.:0.0000
##  Median :0.00   Median :0.00    Median :0.0000
Median :0.0000
##  Mean   :0.25   Mean   :0.25    Mean   :0.3333
Mean   :0.3333
##  3rd Qu.:0.25   3rd Qu.:0.25    3rd Qu.:1.0000
3rd Qu.:1.0000
##  Max.   :1.00   Max.   :1.00    Max.   :1.0000
Max.   :1.0000
##   Contact_Yes       Country
##  Min.   :0.0000   Length:316800
##  1st Qu.:0.0000   Class :character
##  Median :0.0000   Mode  :character
##  Mean   :0.3333
##  3rd Qu.:1.0000
##  Max.   :1.0000
```

```
Cleaned_Data_1 <- Cleaned.Data[1:10000,]
attach(Cleaned_Data_1)
plot(Fever,Sore.Throat)
```



```
library(e1071)
svm.fit <- svm(Severity_Mild ~ Fever+Sore.Throat,
data = Cleaned_Data_1, type='C-classification',
kernel='linear', cost=10, scale=FALSE)
plot(svm.fit, Cleaned_Data_1)
## Error in plot.svm(svm.fit, Cleaned_Data_1):
missing formula.
summary(svm.fit)
##
## Call:
## svm(formula = Severity_Mild ~ Fever +
Sore.Throat, data = Cleaned_Data_1,
##     type = "C-classification", kernel = "linear",
cost = 10, scale = FALSE)
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  linear
##        cost:  10
##
## Number of Support Vectors:  5004
##
##  ( 2502 2502 )
##
```

```
##
## Number of Classes:  2
##
## Levels:
##  0 1
yhat <- predict(svm.fit, Cleaned_Data_1)
table(predict=yhat, truth=Severity_Mild)
##        truth
## predict    0    1
##       0 7498 2502
##       1    0    0
library(caret)
confusionMatrix(yhat,
as.factor(Cleaned_Data_1$Severity_Mild))
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 7498 2502
##          1    0    0
##
##                Accuracy : 0.7498
##                  95% CI : (0.7412, 0.7583)
##     No Information Rate : 0.7498
##     P-Value [Acc > NIR] : 0.5054
##
##                   Kappa : 0
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 1.0000
##             Specificity : 0.0000
##          Pos Pred Value : 0.7498
##          Neg Pred Value :    NaN
##              Prevalence : 0.7498
##          Detection Rate : 0.7498
##    Detection Prevalence : 1.0000
##       Balanced Accuracy : 0.5000
##
##        'Positive' Class : 0
##
```

Lab Exercise No : 9
Exercise Title : Radom Forest Classifier
Date : 26-04-2021

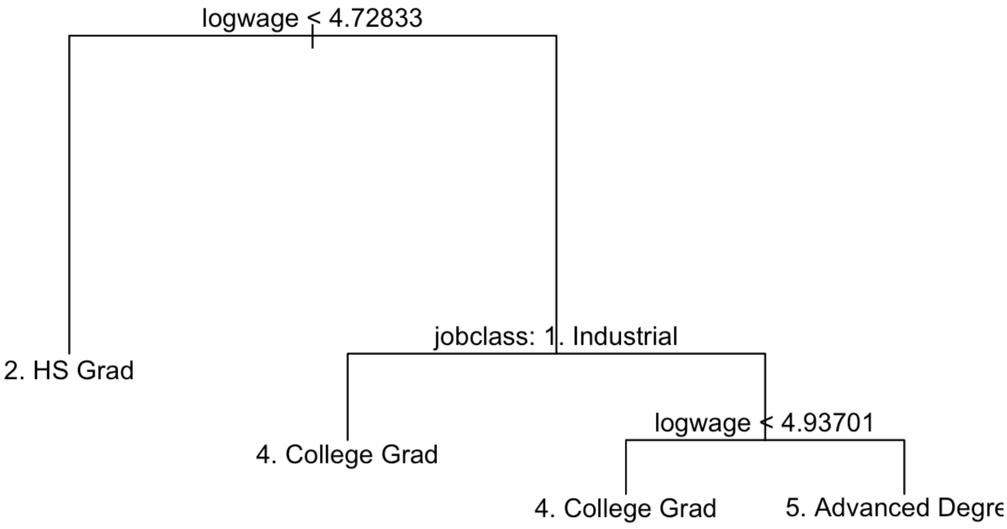
Task:

Develop a Random Forest Classifier for the given Dataset

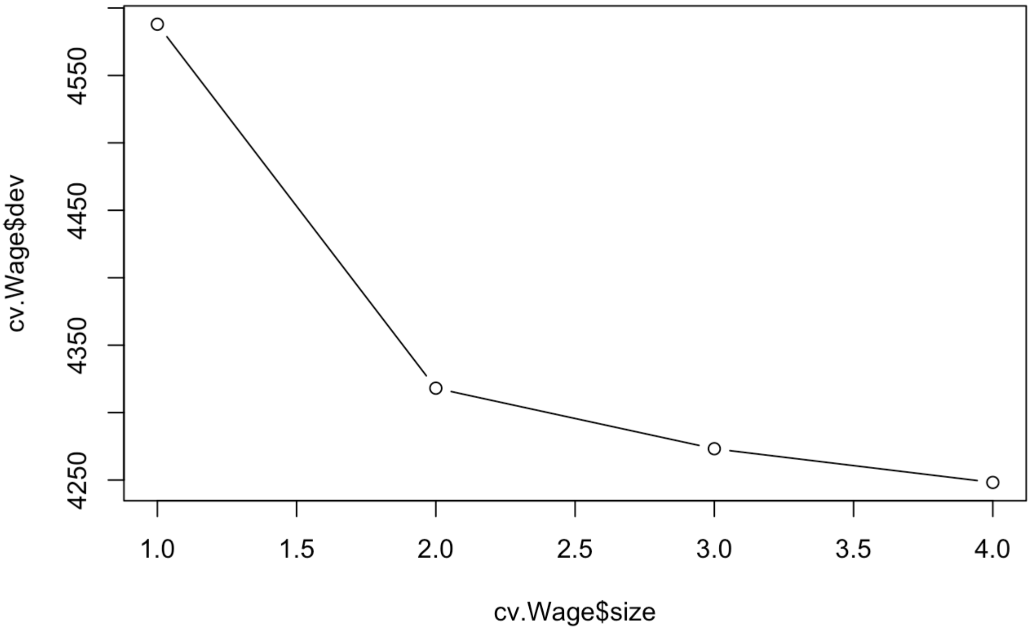
Program:

Output:

```
# Fitting Regression Trees
library(tree)
library(ISLR)
library(MASS)
set.seed(1)
train = sample(1:nrow(Wage), nrow(Wage)/2)
tree.Wage=tree(education~.,Wage,subset=train)
summary(tree.Wage)
##
## Classification tree:
## tree(formula = education ~ ., data = Wage, subset
= train)
## Variables actually used in tree construction:
## [1] "logwage"  "jobclass"
## Number of terminal nodes:  4
## Residual mean deviance:  2.789 = 4172 / 1496
## Misclassification error rate: 0.6073 = 911 / 1500
plot(tree.Wage)
text(tree.Wage,pretty=0)
```

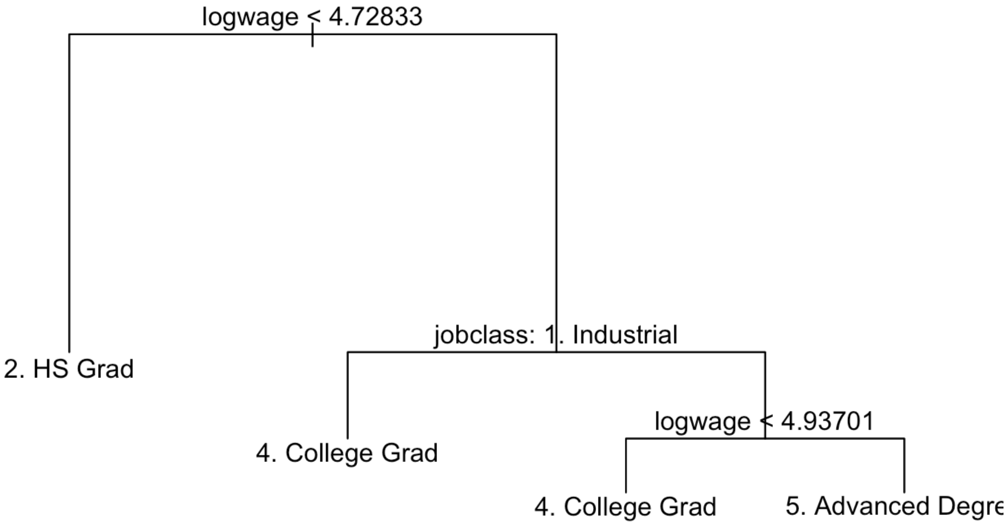


```
cv.Wage=cv.tree(tree.Wage)
plot(cv.Wage$size,cv.Wage$dev,type='b')
```

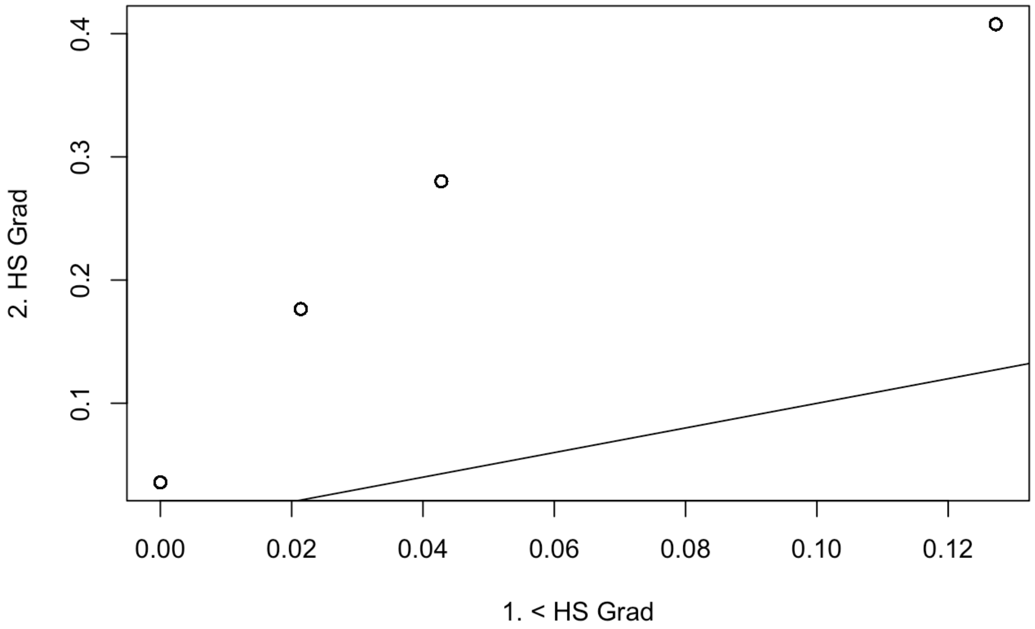


```
prune.Wage=prune.tree(tree.Wage,best=5)
## Warning in prune.tree(tree.Wage, best = 5): best
is bigger than tree size
plot(prune.Wage)
```

```
text(prune.Wage,pretty=0)
```

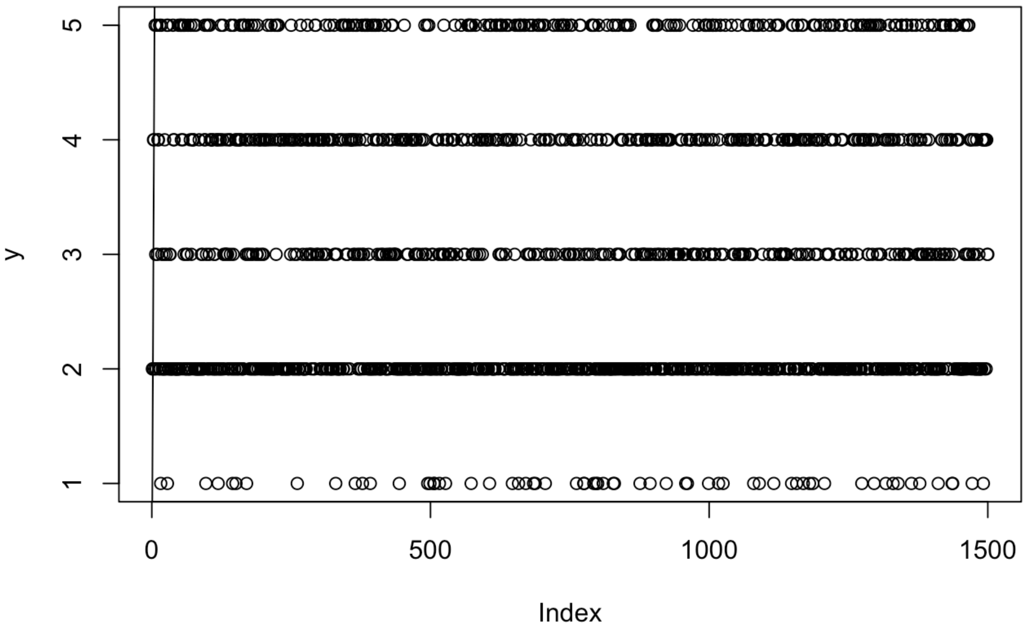


```
yhat=predict(tree.Wage,newdata=Wage[-train,])
Wage.test=Wage[-train,"Education"]
plot(yhat,Wage.test)
abline(0,1)
```



```
mean((yhat-Wage.test)^2)
```

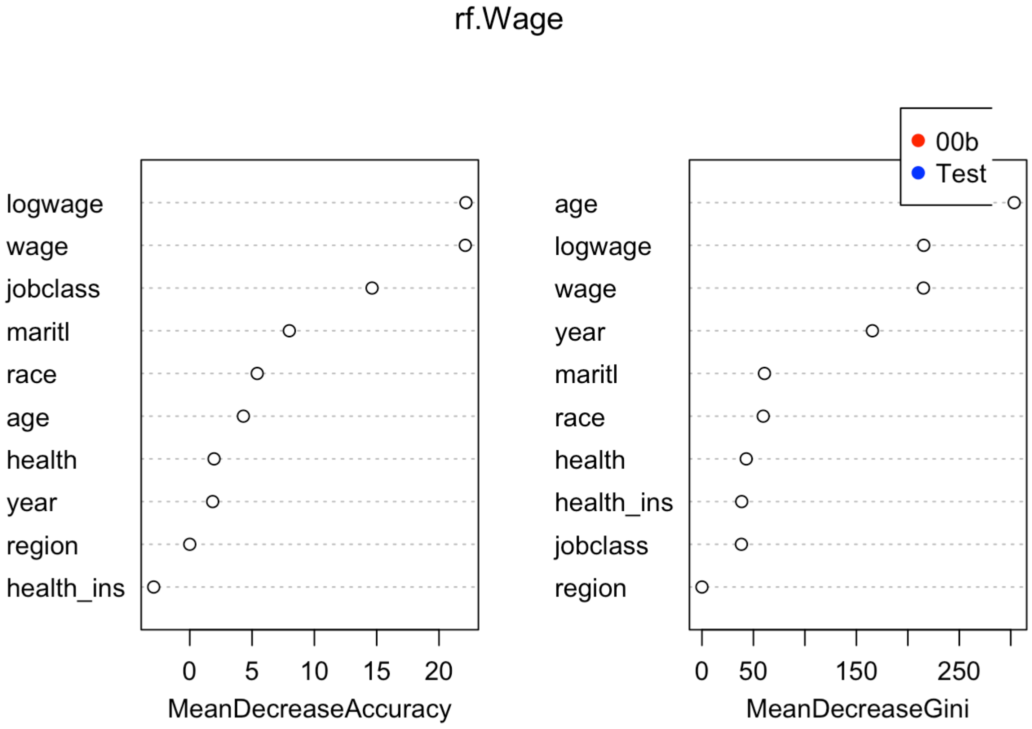
```
## [1] NaN
# Bagging and Random Forests

library(randomForest)
## randomForest 4.6-14
## Type rfNews() to see new features/changes/bug
fixes.
set.seed(1)
bag.Wage=randomForest(education~.,data=Wage,subset=t
rain,mtry=13,importance=TRUE)
## Warning in randomForest.default(m, y, ...):
invalid mtry: reset to within valid
## range
bag.Wage
##
## Call:
##  randomForest(formula = education ~ ., data =
Wage, mtry = 13,      importance = TRUE, subset =
train)
##                Type of random forest:
classification
##                      Number of trees: 500
## No. of variables tried at each split: 10
##
##         OOB estimate of  error rate: 67.8%
## Confusion matrix:
##                      1. < HS Grad 2. HS Grad 3.
Some College 4. College Grad
## 1. < HS Grad                  10         84
21             12
## 2. HS Grad                    27        241
93             91
## 3. Some College               11        139
64             75
## 4. College Grad                9        129
66             91
## 5. Advanced Degree             0         32
32             74
##                      5. Advanced Degree class.error
## 1. < HS Grad                          1   0.9218750
## 2. HS Grad                           21   0.4904863
## 3. Some College                      33   0.8012422
## 4. College Grad                      67   0.7486188
## 5. Advanced Degree                     77    0.6418605
yhat.bag = predict(bag.Wage,newdata=Wage[-train,])
plot(yhat.bag, Wage.test)
abline(0,1)
```



```
mean((yhat.bag-Wage.test)^2)
## Warning in Ops.factor(yhat.bag, Wage.test): '-'
not meaningful for factors
## [1] NA
bag.Wage=randomForest(education~.,data=Wage,subset=t
rain,mtry=13,ntree=25)
## Warning in randomForest.default(m, y, ...):
invalid mtry: reset to within valid
## range
yhat.bag = predict(bag.Wage,newdata=Wage[-train,])
mean((yhat.bag-Wage.test)^2)
## Warning in Ops.factor(yhat.bag, Wage.test): '-'
not meaningful for factors
## [1] NA
set.seed(1)
rf.Wage=randomForest(education~.,data=Wage,subset=tr
ain,mtry=6,importance=TRUE)
yhat.rf = predict(rf.Wage,newdata=Wage[-train,])
mean((yhat.rf-Wage.test)^2)
## Warning in Ops.factor(yhat.rf, Wage.test): '-'
not meaningful for factors
```

```
## [1] NA
importance(rf.Wage)
##              1. < HS Grad 2. HS Grad 3. Some
College 4. College Grad
## year       -0.0793252614  0.3845728
0.0778268       2.0341287
## age         0.9594635455  1.7519811
1.0537875      -0.6432954
## maritl      3.4061761769  6.9322919
3.6131096       2.0988452
## race       -0.0816198690  4.6241024
2.9291564      -0.4524506
## region      0.0000000000  0.0000000
0.0000000       0.0000000
## jobclass    1.9668960509  2.8529161
-4.5109726        4.8465867
## health     -0.0001129919  5.1131494
-3.7068436       -0.5347277
## health_ins  6.7698701377  1.2329665
-6.7815167       -7.6505749
## logwage     8.5617389326 12.1742357
-0.9647908        0.9403824
## wage        8.4399177692 11.2533127
-0.6014296        1.1757907
##            5. Advanced Degree
MeanDecreaseAccuracy MeanDecreaseGini
## year                     1.785247
1.837703        165.59534
## age                      7.544974
4.303833        303.33060
## maritl                  -1.199283
7.990316         60.73061
## race                     3.585829
5.412398         59.54323
## region                   0.000000
0.000000          0.00000
## jobclass                27.739871
14.628208         38.32821
## health                   1.689627
1.951308         43.01624
## health_ins               1.932245
-2.892992          38.60155
```

```
## logwage             24.648261
22.159665        215.46698
## wage                23.357716
22.109760        215.21954
varImpPlot(rf.Wage)

oob.err=double(13)
test.err=double(13)
for (mtry in 1:13){
  fit=randomForest(education~.,data =
Wage,subset=train,mtry=mtry,ntree=400)
  oob.err[mtry]=fit$mse[400]
  pred=predict(fit,Wage[-train,])
  test.err[mtry]=with(Wage[-train,],mean((education-
age)^2))
  cat(mtry," ")
}
## Error in oob.err[mtry] <- fit$mse[400]:
replacement has length zero
matplot(1:mtry,cbind(test.err,oob.err),pch=19,col=c(
"red","blue"),type = "b",ylab="Mean Squared Error")
## Error in matplot(1:mtry, cbind(test.err,
oob.err), pch = 19, col = c("red", : 'x' and 'y'
must have same number of rows
legend("topright",legend=c("00b","Test"),pch=19,col=
c("red","blue"))
```


rf.Wage
logwage
wage
jobclass
maritl
race
age
health
year
region
health_ins
0
5
10
15
20
MeanDecreaseAccuracy
age
logwage
wage
year
maritl
race
health
health_ins
jobclass
region
0
50
150
250
MeanDecreaseGini
● 00b
● Test

Lab Exercise No : 10
Exercise Title : Spline Curve Generation
Date : 17-05-2021

Task:

Generate a Spline Curve Model for the given dataset

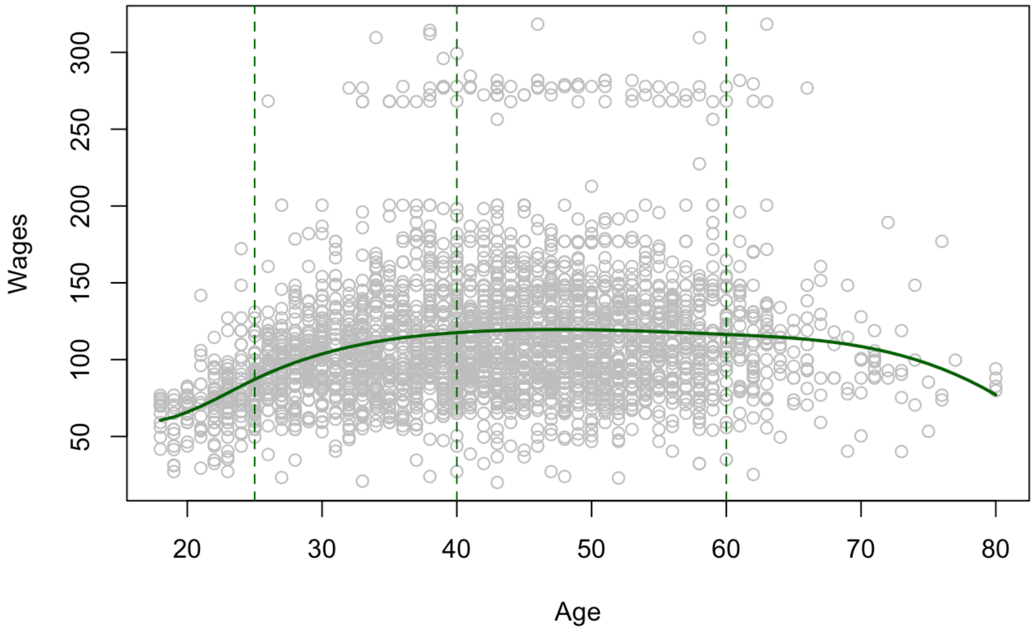
Program:

Output:

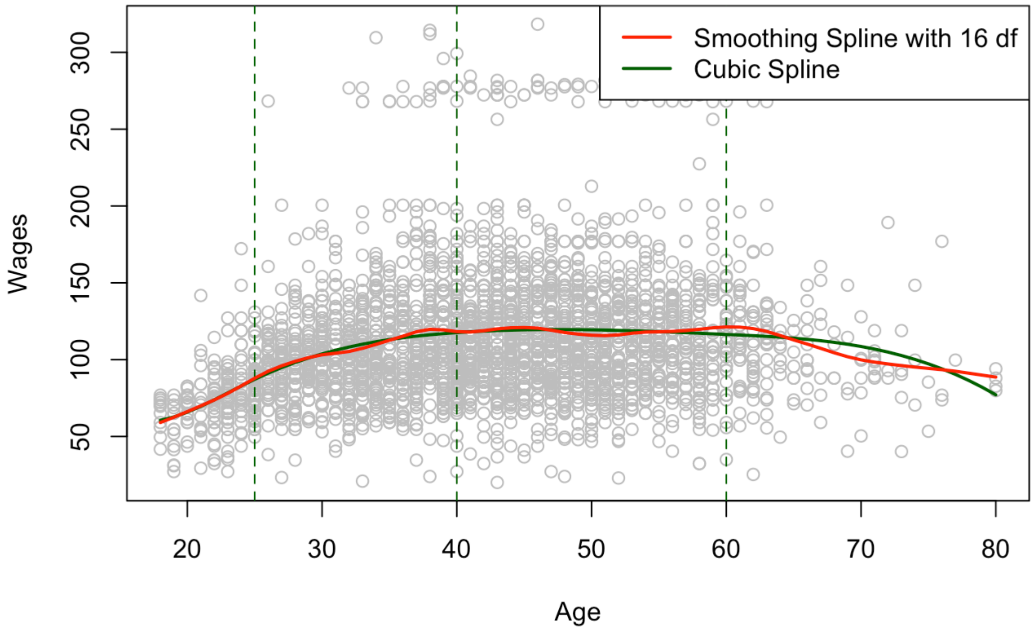
```
library(tidyverse)
## ── Attaching packages
───────────────────────────────────────── tidyverse
1.3.0 ──
## ✓ ggplot2 3.3.3     ✓ purrr   0.3.4
## ✓ tibble  3.1.0     ✓ dplyr   1.0.5
## ✓ tidyr   1.1.3     ✓ stringr 1.4.0
## ✓ readr   1.4.0     ✓ forcats 0.5.1
## ── Conflicts
──────────────────────────────────────────
tidyverse_conflicts() ──
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
library(broom)
require(splines)
## Loading required package: splines
library(ISLR)
attach(Wage)
agelims<-range(Wage$age)
age.grid<-seq(from=agelims[1], to = agelims[2])

fit<-lm(wage ~ bs(age,knots = c(25,40,60)),data =
Wage )
summary(fit)
##
## Call:
## lm(formula = wage ~ bs(age, knots = c(25, 40,
60)), data = Wage)
```

```
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -98.832 -24.537  -5.049  15.209 203.207
##
## Coefficients:
##                                 Estimate Std.
Error t value Pr(>|t|)
## (Intercept)                       60.494
9.460   6.394 1.86e-10 ***
## bs(age, knots = c(25, 40, 60))1    3.980
12.538   0.317 0.750899
## bs(age, knots = c(25, 40, 60))2   44.631
9.626   4.636 3.70e-06 ***
## bs(age, knots = c(25, 40, 60))3   62.839
10.755   5.843 5.69e-09 ***
## bs(age, knots = c(25, 40, 60))4   55.991
10.706   5.230 1.81e-07 ***
## bs(age, knots = c(25, 40, 60))5   50.688
14.402   3.520 0.000439 ***
## bs(age, knots = c(25, 40, 60))6   16.606
19.126   0.868 0.385338
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05
'.' 0.1 ' ' 1
##
## Residual standard error: 39.92 on 2993 degrees of
freedom
## Multiple R-squared:  0.08642,    Adjusted R-
squared:  0.08459
## F-statistic: 47.19 on 6 and 2993 DF,  p-value: <
2.2e-16
```

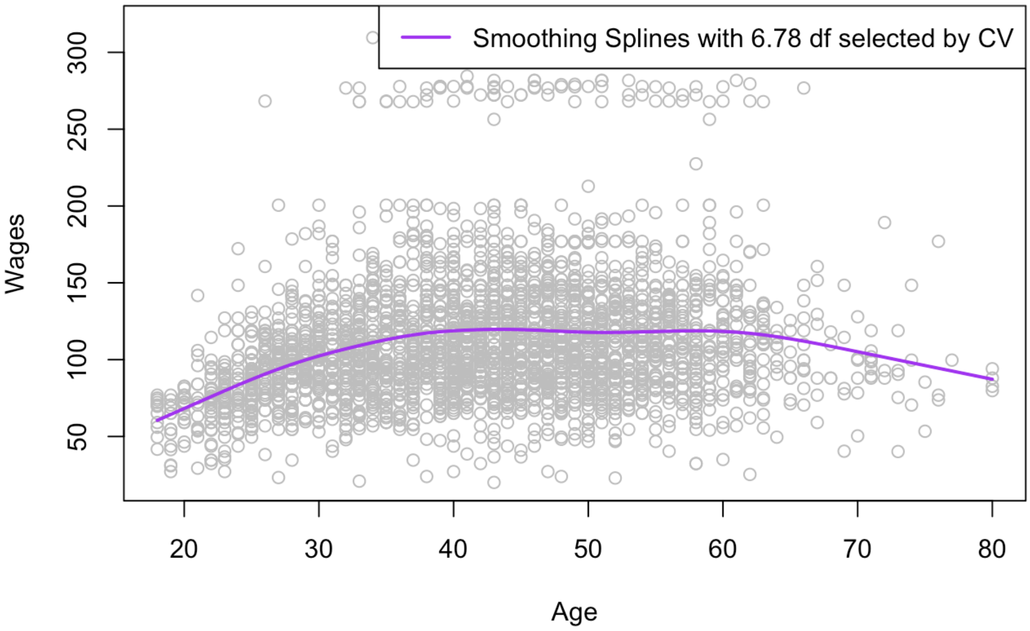
```
plot(Wage$age,Wage$wage,col="grey",xlab="Age",ylab="
Wages")
points(age.grid,predict(fit,newdata =
list(age=age.grid)),col="darkgreen",lwd=2,type="l")

abline(v=c(25,40,60),lty=2,col="darkgreen")
```



```
fit1<-smooth.spline(Wage$age,Wage$wage,df=16) #16
degrees of freedom

plot(Wage$age,Wage$wage,col="grey",xlab="Age",ylab="
Wages")
points(age.grid,predict(fit,newdata =
list(age=age.grid)),col="darkgreen",lwd=2,type="l")

abline(v=c(25,40,60),lty=2,col="darkgreen")
lines(fit1,col="red",lwd=2)
legend("topright",c("Smoothing Spline with 16
df","Cubic Spline"),col=c("red","darkgreen"),lwd=2)
```



```
fit2<-smooth.spline(Wage$age,Wage$wage,cv = TRUE)
## Warning in smooth.spline(Wage$age, Wage$wage, cv
= TRUE): cross-validation with
## non-unique 'x' values seems doubtful
fit2
## Call:
## smooth.spline(x = Wage$age, y = Wage$wage, cv =
TRUE)
##
## Smoothing Parameter  spar= 0.6988943  lambda=
0.02792303 (12 iterations)
## Equivalent Degrees of Freedom (Df): 6.794596
## Penalized Criterion (RSS): 75215.9
## PRESS(l.o.o. CV): 1593.383
plot(Wage$age,Wage$wage,col="grey",xlab="Age",ylab="
Wages")

lines(fit2,lwd=2,col="purple")
legend("topright",("Smoothing Splines with 6.78 df
selected by CV"),col="purple",lwd=2)
```


Smoothing Splines with 6.78 df selected by CV
Wages
50
100
150
200
250
300
20
30
40
50
60
70
80
Age