HW3

December 12, 2021

[148]:
```
library(tidyverse)
install.packages('caret')
install.packages("glmnet", repos = "https://cran.us.r-project.org")
library(caret)
require(gh)
library(stringr)
tmp = tempfile()
qurl = 'https://raw.githubusercontent.com/Nakul24-1/ML-Cars/main/mushrooms.csv'
gh(paste0('GET ', qurl), .destfile = tmp, .overwrite = TRUE)
```

```
Installing package into '/usr/local/lib/R/site-library'
(as 'lib' is unspecified)

Installing package into '/usr/local/lib/R/site-library'
(as 'lib' is unspecified)


[1] "/tmp/RtmpGfY40m/file41741a35ab"
attr(,"class")
[1] "gh_response" "path"
```

[149]:
```
library(rpart)
```

[171]:
```
mush = read.csv(tmp,stringsAsFactors = T)
head(mush)
mush$veil.type
```

A data.frame: 6 × 23

	class <fct>	cap.shape <fct>	cap.surface <fct>	cap.color <fct>	bruises <fct>	odor <fct>	gill.attachment <fct>	gill.sp <fct>
1	p	x	s	n	t	p	f	c
2	e	x	s	y	t	a	f	c
3	e	b	s	w	t	l	f	c
4	p	x	y	w	t	p	f	c
5	e	x	s	g	f	n	f	w
6	e	x	y	y	t	a	f	c

1. p 2. p 3. p 4. p 5. p 6. p 7. p 8. p 9. p 10. p 11. p 12. p 13. p 14. p 15. p 16. p 17. p 18. p 19. p
20. p 21. p 22. p 23. p 24. p 25. p 26. p 27. p 28. p 29. p 30. p 31. p 32. p 33. p 34. p 35. p 36. p
37. p 38. p 39. p 40. p 41. p 42. p 43. p 44. p 45. p 46. p 47. p 48. p 49. p 50. p 51. p 52. p 53. p

54. p 55. p 56. p 57. p 58. p 59. p 60. p 61. p 62. p 63. p 64. p 65. p 66. p 67. p 68. p 69. p 70. p 71. p 72. p 73. p 74. p 75. p 76. p 77. p 78. p 79. p 80. p 81. p 82. p 83. p 84. p 85. p 86. p 87. p 88. p 89. p 90. p 91. p 92. p 93. p 94. p 95. p 96. p 97. p 98. p 99. p 100. p 101. p 102. p 103. p 104. p 105. p 106. p 107. p 108. p 109. p 110. p 111. p 112. p 113. p 114. p 115. p 116. p 117. p 118. p 119. p 120. p 121. p 122. p 123. p 124. p 125. p 126. p 127. p 128. p 129. p 130. p 131. p 132. p 133. p 134. p 135. p 136. p 137. p 138. p 139. p 140. p 141. p 142. p 143. p 144. p 145. p 146. p 147. p 148. p 149. p 150. p 151. p 152. p 153. p 154. p 155. p 156. p 157. p 158. p 159. p 160. p 161. p 162. p 163. p 164. p 165. p 166. p 167. p 168. p 169. p 170. p 171. p 172. p 173. p 174. p 175. p 176. p 177. p 178. p 179. p 180. p 181. p 182. p 183. p 184. p 185. p 186. p 187. p 188. p 189. p 190. p 191. p 192. p 193. p 194. p 195. p 196. p 197. p 198. p 199. p 200. p 201. 202. p 203. p 204. p 205. p 206. p 207. p 208. p 209. p 210. p 211. p 212. p 213. p 214. p 215. p 216. p 217. p 218. p 219. p 220. p 221. p 222. p 223. p 224. p 225. p 226. p 227. p 228. p 229. p 230. p 231. p 232. p 233. p 234. p 235. p 236. p 237. p 238. p 239. p 240. p 241. p 242. p 243. p 244. p 245. p 246. p 247. p 248. p 249. p 250. p 251. p 252. p 253. p 254. p 255. p 256. p 257. p 258. p 259. p 260. p 261. p 262. p 263. p 264. p 265. p 266. p 267. p 268. p 269. p 270. p 271. p 272. p 273. p 274. p 275. p 276. p 277. p 278. p 279. p 280. p 281. p 282. p 283. p 284. p 285. p 286. p 287. p 288. p 289. p 290. p 291. p 292. p 293. p 294. p 295. p 296. p 297. p 298. p 299. p 300. p 301. p 302. p 303. p 304. p 305. p 306. p 307. p 308. p 309. p 310. p 311. p 312. p 313. p 314. p 315. p 316. p 317. p 318. p 319. p 320. p 321. p 322. p 323. p 324. p 325. p 326. p 327. p 328. p 329. p 330. p 331. p 332. p 333. p 334. p 335. p 336. p 337. p 338. p 339. p 340. p 341. p 342. p 343. p 344. p 345. p 346. p 347. p 348. p 349. p 350. p 351. p 352. p 353. p 354. p 355. p 356. p 357. p 358. p 359. p 360. p 361. p 362. p 363. p 364. p 365. p 366. p 367. p 368. p 369. p 370. p 371. p 372. p 373. p 374. p 375. p 376. p 377. p 378. p 379. p 380. p 381. p 382. p 383. p 384. p 385. p 386. p 387. p 388. p 389. p 390. p 391. p 392. p 393. p 394. p 395. p 396. p 397. p 398. p 399. p 400. p 401. p

Levels: 'p'

[172]:

```
set.seed(121)

mush = mush %>% select(-veil.type)
mush_x = mush %>% select(-class)
mush_y = mush %>% select(class)

size<- floor(0.7*nrow(mush))
train_ind <- sample(seq_len(nrow(mush)), size = size)
train<-mush[train_ind,]
test<-mush[-train_ind,]
train_y <- as.data.frame(mush_y[train_ind,])
test_y<-as.data.frame(mush_y[-train_ind,])
names(train_y) = 'class'
names(test_y) = 'class'
true_test_y = 1*(test$class == 'p')
true_test_y
```

1. 0 2. 1 3. 0 4. 0 5. 0 6. 0 7. 0 8. 0 9. 0 10. 0 11. 0 12. 0 13. 0 14. 1 15. 0 16. 0 17. 0 18. 0 19. 0 20. 0 21. 0 22. 0 23. 0 24. 1 25. 0 26. 0 27. 0 28. 0 29. 1 30. 0 31. 0 32. 0 33. 0 34. 0 35. 0 36. 0 37. 0 38. 0 39. 0 40. 1 41. 0 42. 0 43. 0 44. 0 45. 0 46. 0 47. 0 48. 0 49. 1 50. 0 51. 0 52. 0 53. 0 54. 0 55. 0 56. 0 57. 1 58. 0 59. 0 60. 0 61. 0 62. 0 63. 0 64. 0 65. 0 66. 0 67. 0 68. 0 69. 0 70. 0 71. 1 72. 0 73. 0

74. 0 75. 1 76. 0 77. 0 78. 0 79. 0 80. 0 81. 0 82. 0 83. 0 84. 0 85. 0 86. 0 87. 0 88. 0 89. 0 90. 0 91. 0 92. 0 93. 1 94. 0 95. 0 96. 0 97. 0 98. 0 99. 0 100. 0 101. 0 102. 0 103. 0 104. 0 105. 0 106. 0 107. 0 108. 0 109. 1 110. 0 111. 0 112. 0 113. 0 114. 0 115. 0 116. 0 117. 1 118. 0 119. 0 120. 0 121. 0 122. 0 123. 0 124. 0 125. 0 126. 0 127. 0 128. 0 129. 0 130. 0 131. 0 132. 0 133. 0 134. 0 135. 0 136. 0 137. 0 138. 0 139. 0 140. 0 141. 0 142. 0 143. 0 144. 0 145. 0 146. 0 147. 0 148. 0 149. 0 150. 0 151. 1 152. 0 153. 0 154. 0 155. 0 156. 0 157. 0 158. 0 159. 0 160. 0 161. 1 162. 0 163. 0 164. 0 165. 1 166. 0 167. 0 168. 0 169. 0 170. 0 171. 0 172. 0 173. 1 174. 0 175. 1 176. 0 177. 0 178. 0 179. 0 180. 1 181. 0 182. 0 183. 0 184. 0 185. 0 186. 0 187. 0 188. 0 189. 0 190. 0 191. 0 192. 0 193. 0 194. 0 195. 0 196. 0 197. 0 198. 0 199. 0 200. 0 201. 202. 0 203. 0 204. 0 205. 1 206. 1 207. 1 208. 1 209. 1 210. 0 211. 1 212. 1 213. 1 214. 1 215. 1 216. 1 217. 1 218. 1 219. 1 220. 1 221. 1 222. 0 223. 1 224. 1 225. 0 226. 1 227. 1 228. 0 229. 0 230. 0 231. 0 232. 0 233. 0 234. 1 235. 1 236. 1 237. 1 238. 0 239. 0 240. 1 241. 0 242. 1 243. 1 244. 0 245. 1 246. 1 247. 1 248. 1 249. 1 250. 1 251. 0 252. 1 253. 1 254. 0 255. 1 256. 1 257. 0 258. 0 259. 1 260. 0 261. 0 262. 1 263. 1 264. 0 265. 1 266. 1 267. 1 268. 1 269. 0 270. 1 271. 0 272. 1 273. 1 274. 1 275. 0 276. 1 277. 1 278. 1 279. 0 280. 0 281. 1 282. 0 283. 0 284. 1 285. 1 286. 0 287. 1 288. 0 289. 0 290. 0 291. 1 292. 1 293. 1 294. 1 295. 1 296. 0 297. 1 298. 1 299. 0 300. 0 301. 0 302. 0 303. 1 304. 1 305. 0 306. 0 307. 0 308. 0 309. 1 310. 1 311. 0 312. 0 313. 1 314. 0 315. 1 316. 0 317. 1 318. 0 319. 1 320. 1 321. 0 322. 0 323. 1 324. 0 325. 1 326. 0 327. 0 328. 1 329. 0 330. 0 331. 1 332. 1 333. 1 334. 0 335. 0 336. 0 337. 1 338. 0 339. 0 340. 1 341. 0 342. 1 343. 0 344. 1 345. 0 346. 1 347. 0 348. 0 349. 1 350. 1 351. 1 352. 1 353. 1 354. 1 355. 0 356. 1 357. 0 358. 1 359. 0 360. 0 361. 0 362. 0 363. 0 364. 0 365. 0 366. 1 367. 1 368. 1 369. 0 370. 1 371. 0 372. 1 373. 1 374. 1 375. 0 376. 0 377. 1 378. 0 379. 0 380. 1 381. 0 382. 1 383. 1 384. 0 385. 0 386. 1 387. 0 388. 0 389. 0 390. 1 391. 1 392. 0 393. 1 394. 0 395. 0 396. 0 397. 0 398. 0 399. 0 400. 1 401. 0

1 Random Forest

[173]:
```
install.packages('doParallel')
install.packages('randomForest')
library(doParallel)
library(future)
```

```
Installing package into '/usr/local/lib/R/site-library'
(as 'lib' is unspecified)

Installing package into '/usr/local/lib/R/site-library'
(as 'lib' is unspecified)
```

[174]:
```
library(randomForest)
        classifier_rf = randomForest(x = train[-1], y = train$class,
                                     data = train,ntree = 100)
        y_pred_rf = predict(classifier_rf, newdata = test[-1])
        confusionMatrix(test$class, y_pred_rf)
```

```
Confusion Matrix and Statistics

          Reference
Prediction    e    p
         e 1283    0
```

```
         p    0 1155

               Accuracy : 1
                 95% CI : (0.9985, 1)
    No Information Rate : 0.5263
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 1

 Mcnemar's Test P-Value : NA

            Sensitivity : 1.0000
            Specificity : 1.0000
         Pos Pred Value : 1.0000
         Neg Pred Value : 1.0000
             Prevalence : 0.5263
         Detection Rate : 0.5263
   Detection Prevalence : 0.5263
      Balanced Accuracy : 1.0000

       'Positive' Class : e
```

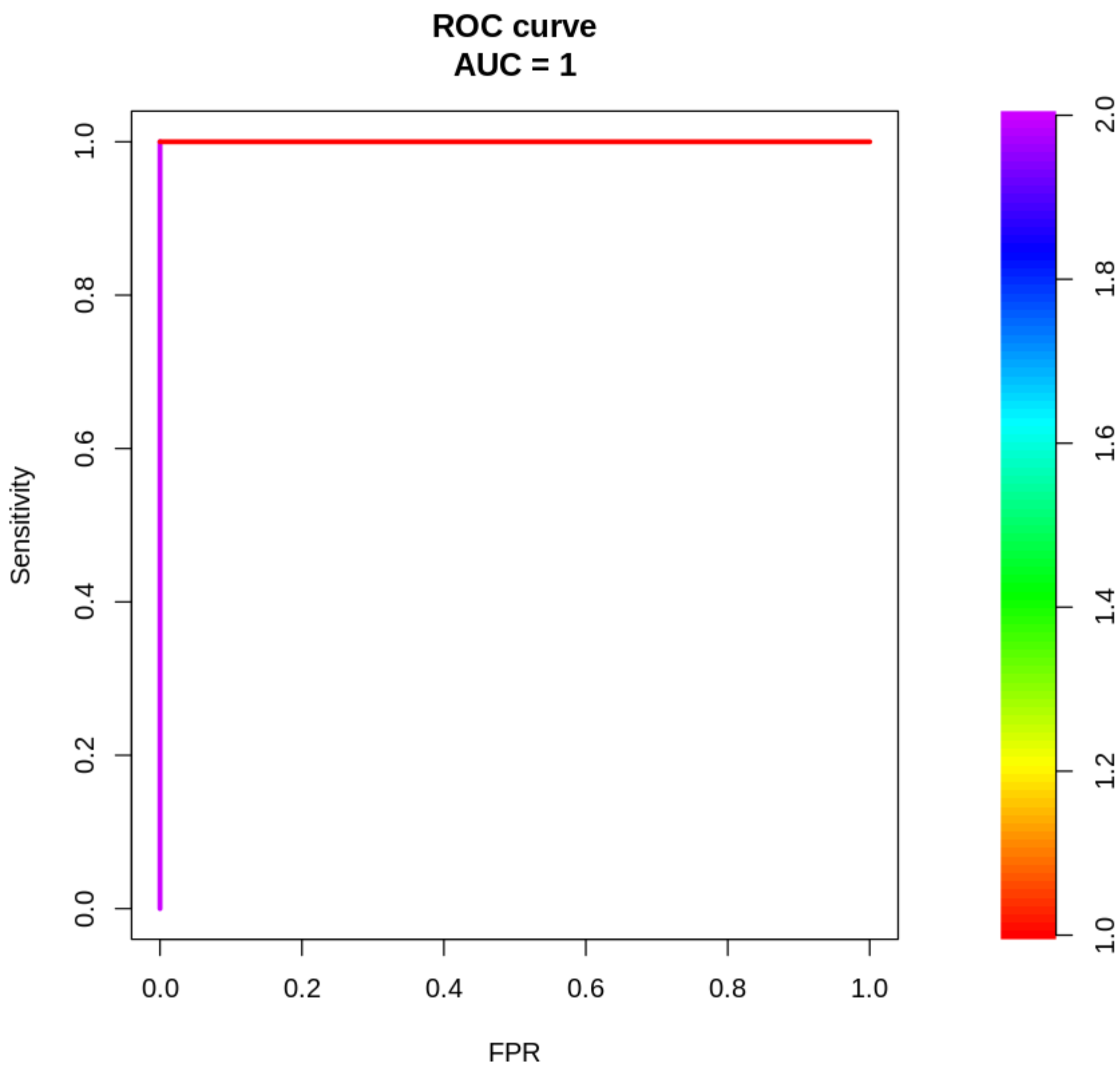
[167]:
```
install.packages('PRROC')
library(PRROC)
```

```
Installing package into '/usr/local/lib/R/site-library'
(as 'lib' is unspecified)
```

```
    Error in `[.data.frame`(weights.class0, o0): undefined columns selected
Traceback:

    1. roc.curve(scores.class0 = y_pred_rf, weights.class0 = test_y,
 .     curve = TRUE)

    2. weights.class0[o0]

    3. `[.data.frame`(weights.class0, o0)

    4. stop("undefined columns selected")
```

[175]:
```
PRROC_obj <- roc.curve(scores.class0 = y_pred_rf, weights.class0 = true_test_y,
                       curve=TRUE)
```

```
plot(PRROC_obj)
```



2 AdaBoost

[154]:
```
install.packages('fastAdaboost')
library(fastAdaboost)
```

```
Installing package into ‘/usr/local/lib/R/site-library’
(as ‘lib’ is unspecified)
```

[190]:
```
ad <- adaboost(class ~., data = train, tree_depth = 5, n_rounds = 5,10)
```

[191]:
```
y_pred_ada = predict(ad, newdata = test[-1])
y_pred_ada$class
```

1. e 2. p 3. e 4. e 5. e 6. e 7. e 8. e 9. e 10. e 11. e 12. e 13. e 14. p 15. e 16. e 17. e 18. e 19. e 20. e 21. e 22. e 23. e 24. p 25. e 26. e 27. e 28. e 29. p 30. e 31. e 32. e 33. e 34. e 35. e 36. e 37. e 38. e 39. e 40. p 41. e 42. e 43. e 44. e 45. e 46. e 47. e 48. e 49. p 50. e 51. e 52. e 53. e 54. e 55. e 56. e 57. p 58. e 59. e 60. e 61. e 62. e 63. e 64. e 65. e 66. e 67. e 68. e 69. e 70. e 71. p 72. e 73. e 74. e 75. p 76. e 77. e 78. e 79. e 80. e 81. e 82. e 83. e 84. e 85. e 86. e 87. e 88. e 89. e 90. e 91. e 92. e 93. p 94. e 95. e 96. e 97. e 98. e 99. e 100. e 101. e 102. e 103. e 104. e 105. e 106. e 107. e 108. e 109. p 110. e 111. e 112. e 113. e 114. e 115. e 116. e 117. p 118. e 119. e 120. e 121. e 122. e 123. e 124. e 125. e 126. e 127. e 128. e 129. e 130. e 131. e 132. e 133. e 134. e 135. e 136. e 137. e 138. e 139. e 140. e 141. e 142. e 143. e 144. e 145. e 146. e 147. e 148. e 149. e 150. e 151. p 152. e 153. e 154. e 155. e 156. e 157. e 158. e 159. e 160. e 161. p 162. e 163. e 164. e 165. p 166. e 167. e 168. e 169. e 170. e 171. e 172. e 173. p 174. e 175. p 176. e 177. e 178. e 179. e 180. p 181. e 182. e 183. e 184. e 185. e 186. e 187. e 188. e 189. e 190. e 191. e 192. e 193. e 194. e 195. e 196. e 197. e 198. e 199. e 200. e 201. 202. e 203. e 204. e 205. p 206. p 207. p 208. p 209. p 210. e 211. p 212. p 213. p 214. p 215. p 216. p 217. p 218. p 219. p 220. p 221. p 222. e 223. p 224. p 225. e 226. p 227. p 228. e 229. e 230. e 231. e 232. e 233. e 234. p 235. p 236. p 237. p 238. e 239. e 240. p 241. e 242. p 243. p 244. e 245. p 246. p 247. p 248. p 249. p 250. p 251. e 252. p 253. p 254. e 255. p 256. p 257. e 258. e 259. p 260. e 261. e 262. p 263. p 264. e 265. p 266. p 267. p 268. p 269. e 270. p 271. e 272. p 273. p 274. p 275. e 276. p 277. p 278. p 279. e 280. e 281. p 282. e 283. e 284. p 285. p 286. e 287. p 288. e 289. e 290. e 291. p 292. p 293. p 294. p 295. p 296. e 297. p 298. p 299. e 300. e 301. e 302. e 303. p 304. p 305. e 306. e 307. e 308. e 309. p 310. p 311. e 312. e 313. p 314. e 315. p 316. e 317. p 318. e 319. p 320. p 321. e 322. e 323. p 324. e 325. p 326. e 327. e 328. p 329. e 330. e 331. p 332. p 333. p 334. e 335. e 336. e 337. p 338. e 339. e 340. p 341. e 342. p 343. e 344. p 345. e 346. p 347. e 348. e 349. p 350. p 351. p 352. p 353. p 354. p 355. e 356. p 357. e 358. p 359. e 360. e 361. e 362. e 363. e 364. e 365. e 366. p 367. p 368. p 369. e 370. p 371. e 372. p 373. p 374. p 375. e 376. e 377. p 378. e 379. e 380. p 381. e 382. p 383. p 384. e 385. e 386. p 387. e 388. e 389. e 390. p 391. p 392. e 393. p 394. e 395. e 396. e 397. e 398. e 399. e 400. p 401. e

Levels: 1. 'e' 2. 'p'

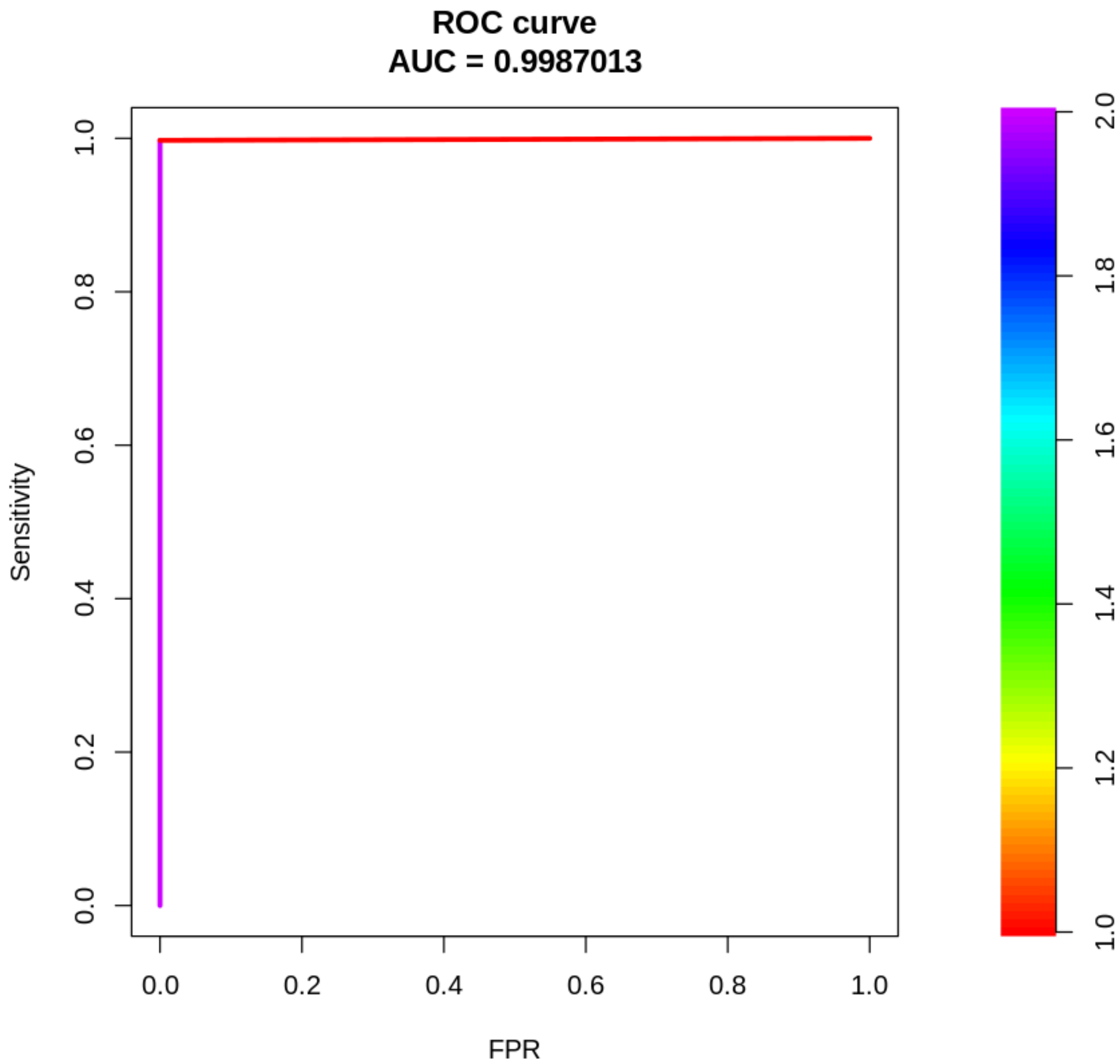
[192]:
```
confusionMatrix(y_pred_ada$class,test$class)
```

```
Confusion Matrix and Statistics

          Reference
Prediction    e    p
         e 1283    3
         p    0 1152

               Accuracy : 0.9988
                 95% CI : (0.9964, 0.9997)
    No Information Rate : 0.5263
    P-Value [Acc > NIR] : <2e-16

                  Kappa : 0.9975

 Mcnemar's Test P-Value : 0.2482
```

```
            Sensitivity : 1.0000
            Specificity : 0.9974
         Pos Pred Value : 0.9977
         Neg Pred Value : 1.0000
             Prevalence : 0.5263
         Detection Rate : 0.5263
   Detection Prevalence : 0.5275
      Balanced Accuracy : 0.9987

       'Positive' Class : e
```

[193]:

```
PRROC_obj <- roc.curve(scores.class0 = as.factor(y_pred_ada$class) , weights.
↪class0 = true_test_y,
                       curve=TRUE)
plot(PRROC_obj)
```



3 Bagging

[194]:
```
bag <- bagging(class ~., data = train,30)
```

[195]:
```
y_pred_bag = predict(bag, newdata = test[-1])
y_pred_bag$class
```

1. 'e' 2. 'p' 3. 'e' 4. 'e' 5. 'e' 6. 'e' 7. 'e' 8. 'e' 9. 'e' 10. 'e' 11. 'e' 12. 'e' 13. 'e' 14. 'p' 15. 'e' 16. 'e' 17. 'e' 18. 'e' 19. 'e' 20. 'e' 21. 'e' 22. 'e' 23. 'e' 24. 'p' 25. 'e' 26. 'e' 27. 'e' 28. 'e' 29. 'p' 30. 'e' 31. 'e' 32. 'e' 33. 'e' 34. 'e' 35. 'e' 36. 'e' 37. 'e' 38. 'e' 39. 'e' 40. 'p' 41. 'e' 42. 'e' 43. 'e' 44. 'e' 45. 'e' 46. 'e' 47. 'e' 48. 'e' 49. 'p' 50. 'e' 51. 'e' 52. 'e' 53. 'e' 54. 'e' 55. 'e' 56. 'e' 57. 'p' 58. 'e' 59. 'e' 60. 'e' 61. 'e' 62. 'e' 63. 'e' 64. 'e' 65. 'e' 66. 'e' 67. 'e' 68. 'e' 69. 'e' 70. 'e' 71. 'p' 72. 'e' 73. 'e' 74. 'e' 75. 'p' 76. 'e' 77. 'e' 78. 'e' 79. 'e' 80. 'e' 81. 'e' 82. 'e' 83. 'e' 84. 'e' 85. 'e' 86. 'e' 87. 'e' 88. 'e' 89. 'e' 90. 'e'

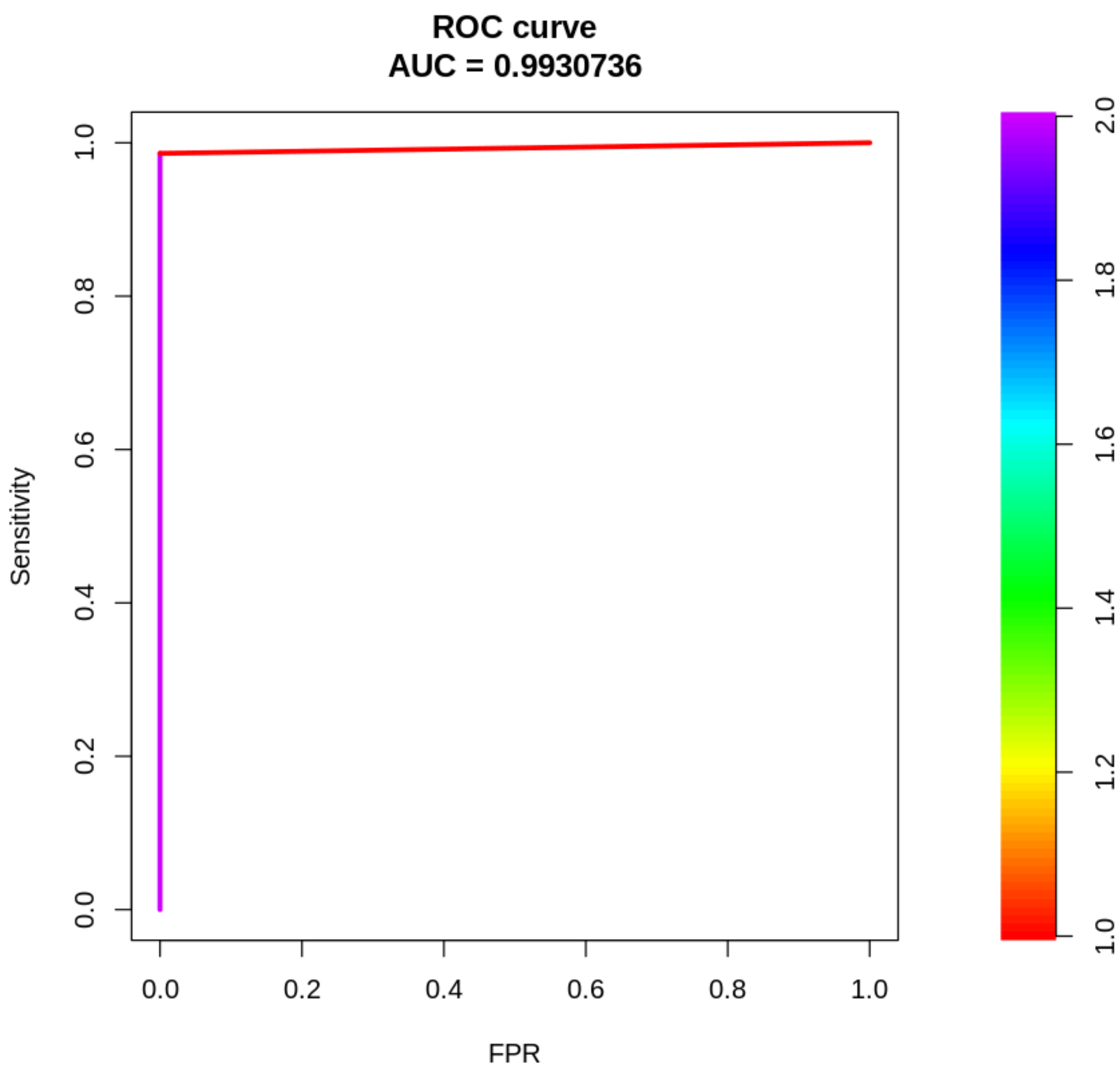
91. 'e' 92. 'e' 93. 'p' 94. 'e' 95. 'e' 96. 'e' 97. 'e' 98. 'e' 99. 'e' 100. 'e' 101. 'e' 102. 'e' 103. 'e' 104. 'e' 105. 'e' 106. 'e' 107. 'e' 108. 'e' 109. 'p' 110. 'e' 111. 'e' 112. 'e' 113. 'e' 114. 'e' 115. 'e' 116. 'e' 117. 'p' 118. 'e' 119. 'e' 120. 'e' 121. 'e' 122. 'e' 123. 'e' 124. 'e' 125. 'e' 126. 'e' 127. 'e' 128. 'e' 129. 'e' 130. 'e' 131. 'e' 132. 'e' 133. 'e' 134. 'e' 135. 'e' 136. 'e' 137. 'e' 138. 'e' 139. 'e' 140. 'e' 141. 'e' 142. 'e' 143. 'e' 144. 'e' 145. 'e' 146. 'e' 147. 'e' 148. 'e' 149. 'e' 150. 'e' 151. 'p' 152. 'e' 153. 'e' 154. 'e' 155. 'e' 156. 'e' 157. 'e' 158. 'e' 159. 'e' 160. 'e' 161. 'p' 162. 'e' 163. 'e' 164. 'e' 165. 'p' 166. 'e' 167. 'e' 168. 'e' 169. 'e' 170. 'e' 171. 'e' 172. 'e' 173. 'p' 174. 'e' 175. 'p' 176. 'e' 177. 'e' 178. 'e' 179. 'e' 180. 'p' 181. 'e' 182. 'e' 183. 'e' 184. 'e' 185. 'e' 186. 'e' 187. 'e' 188. 'e' 189. 'e' 190. 'e' 191. 'e' 192. 'e' 193. 'e' 194. 'e' 195. 'e' 196. 'e' 197. 'e' 198. 'e' 199. 'e' 200. 'e' 201. 202. 'e' 203. 'e' 204. 'e' 205. 'p' 206. 'p' 207. 'p' 208. 'p' 209. 'p' 210. 'e' 211. 'p' 212. 'p' 213. 'p' 214. 'p' 215. 'p' 216. 'p' 217. 'p' 218. 'p' 219. 'p' 220. 'p' 221. 'p' 222. 'e' 223. 'p' 224. 'p' 225. 'e' 226. 'p' 227. 'p' 228. 'e' 229. 'e' 230. 'e' 231. 'e' 232. 'e' 233. 'e' 234. 'p' 235. 'p' 236. 'p' 237. 'p' 238. 'e' 239. 'e' 240. 'p' 241. 'e' 242. 'p' 243. 'p' 244. 'e' 245. 'p' 246. 'p' 247. 'p' 248. 'p' 249. 'p' 250. 'p' 251. 'e' 252. 'p' 253. 'p' 254. 'e' 255. 'p' 256. 'p' 257. 'e' 258. 'e' 259. 'p' 260. 'e' 261. 'e' 262. 'p' 263. 'p' 264. 'e' 265. 'p' 266. 'p' 267. 'p' 268. 'p' 269. 'e' 270. 'p' 271. 'e' 272. 'p' 273. 'p' 274. 'p' 275. 'e' 276. 'p' 277. 'p' 278. 'p' 279. 'e' 280. 'e' 281. 'p' 282. 'e' 283. 'e' 284. 'e' 285. 'p' 286. 'e' 287. 'p' 288. 'e' 289. 'e' 290. 'e' 291. 'p' 292. 'p' 293. 'p' 294. 'p' 295. 'p' 296. 'e' 297. 'p' 298. 'p' 299. 'e' 300. 'e' 301. 'e' 302. 'e' 303. 'p' 304. 'p' 305. 'e' 306. 'e' 307. 'e' 308. 'e' 309. 'p' 310. 'p' 311. 'e' 312. 'e' 313. 'p' 314. 'e' 315. 'p' 316. 'e' 317. 'p' 318. 'e' 319. 'p' 320. 'p' 321. 'e' 322. 'e' 323. 'p' 324. 'e' 325. 'p' 326. 'e' 327. 'e' 328. 'p' 329. 'e' 330. 'e' 331. 'p' 332. 'p' 333. 'p' 334. 'e' 335. 'e' 336. 'e' 337. 'p' 338. 'e' 339. 'e' 340. 'p' 341. 'e' 342. 'p' 343. 'e' 344. 'p' 345. 'e' 346. 'p' 347. 'e' 348. 'e' 349. 'p' 350. 'p' 351. 'p' 352. 'p' 353. 'p' 354. 'p' 355. 'e' 356. 'p' 357. 'e' 358. 'p' 359. 'e' 360. 'e' 361. 'e' 362. 'e' 363. 'e' 364. 'e' 365. 'e' 366. 'p' 367. 'p' 368. 'p' 369. 'e' 370. 'p' 371. 'e' 372. 'p' 373. 'p' 374. 'p' 375. 'e' 376. 'e' 377. 'p' 378. 'e' 379. 'e' 380. 'p' 381. 'e' 382. 'p' 383. 'p' 384. 'e' 385. 'e' 386. 'p' 387. 'e' 388. 'e' 389. 'e' 390. 'p' 391. 'p' 392. 'e' 393. 'p' 394. 'e' 395. 'e' 396. 'e' 397. 'e' 398. 'e' 399. 'e' 400. 'p' 401. 'e'

[196]:
```
confusionMatrix(as.factor(y_pred_bag$class),test$class)
```

```
Confusion Matrix and Statistics

          Reference
Prediction    e    p
         e 1283   16
         p    0 1139

               Accuracy : 0.9934
                 95% CI : (0.9894, 0.9962)
    No Information Rate : 0.5263
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.9868

 Mcnemar's Test P-Value : 0.0001768

            Sensitivity : 1.0000
            Specificity : 0.9861
         Pos Pred Value : 0.9877
```

```
         Neg Pred Value : 1.0000
             Prevalence : 0.5263
         Detection Rate : 0.5263
   Detection Prevalence : 0.5328
      Balanced Accuracy : 0.9931

       'Positive' Class : e
```

[197]:
```
PRROC_obj <- roc.curve(scores.class0 = as.factor(y_pred_bag$class) , weights.
 ↪class0 = true_test_y,
                        curve=TRUE)
plot(PRROC_obj)
```



4 Running in Parallel and comparision

[198]:
```
cl <- makeCluster(3)



setDefaultCluster(cl)
registerDoParallel(cl)
```

[199]:
```
system.time({
classifier_rf = randomForest(x = train[-1], y = train$class,
                                     data = train,ntree = 250)
bag <- bagging(class ~., data = train,35)
ad <- adaboost(class ~., data = train, tree_depth = 10, n_rounds = 5,10)
})
```

```
   user  system elapsed
  5.269   0.178   5.460
```

[188]:
```
stopCluster(cl) # close multi-core cluster
rm(cl)
```

[200]:
```
system.time({
classifier_rf = randomForest(x = train[-1], y = train$class,
                                     data = train,ntree = 250)
bag <- bagging(class ~., data = train,35)
ad <- adaboost(class ~., data = train, tree_depth = 10, n_rounds = 5,10)
  })
```

```
   user  system elapsed
  5.526   0.056   5.594
```

[ ]:
```

```

HW 3
Nakul Pacheriwala (np2455)

Summary of what is done –

Selection of Methods.

Since only decision trees did not provide perfect predictions in the previous report, I have tested on methods which try and improvise decision tree via various ensemble models.

Thus, I tried Boosting, Bagging and Random Forests.

The most improvement was given by Random Forests.

This is observed because multiple random trees are chosen so the issue of overfitting and underfitting both are resolved as more factors are considered while importance to each factor is also reduced.

Boosting made significant improvement even if it didn't make it perfect.

This is because Boosting reduces bias and variance both which was an issue with the original decision tree. The larger number of weak trees and taking average of all produces similar effect as seen in random forests.

Bagging did not make any improvement in our accuracy compared to basic decision tree. This makes sense as bagging is used to reduce variance while keeping trying to preserve bias, while our original tree did not have a high variance as the depth was very less. Thus, is made no improvement.

The bias – variance effect in cv.

In K – Fold cross validation in small datasets–

As we increase K value, both bias and variance are improved till it reaches a threshold

After reaching a threshold value for K, we don't see much improvement.

Also, for very large datasets, K – Fold seems to have lower effect on Bias and Variance.

To determine a classifier's bias is the difference between its averaged estimated and true function, whereas the variance of a classifier is the expected divergence of the estimated prediction function from its average value.

Since the dataset has only have categorical data and it is a classification problem, getting values of Bias and variance would not make a lot of sense as we do not know the true function for finding bias and average of Poison and edible does not make lot of sense.