Ezra Cohen lab 10

August 4, 2021

[1]:

```r
#install.packages("kernlab")
library(kernlab)
#install.packages("caret")
library(caret)
data("GermanCredit")
subCredit <- GermanCredit[,1:10]
str(subCredit)
```

```
Updating HTML index of packages in '.Library'

Making 'packages.html' …
 done

Updating HTML index of packages in '.Library'

Making 'packages.html' …
 done

Loading required package: lattice

Loading required package: ggplot2


Attaching package: ‘ggplot2’


The following object is masked from ‘package:kernlab’:

    alpha



'data.frame':   1000 obs. of  10 variables:
 $ Duration                 : int  6 48 12 42 24 36 24 36 12 30 …
 $ Amount                   : int  1169 5951 2096 7882 4870 9055 2835 6948 3059
5234 …
 $ InstallmentRatePercentage: int  4 2 2 2 3 2 3 2 2 4 …
 $ ResidenceDuration        : int  4 2 3 4 4 4 4 2 4 2 …
 $ Age                       : int  67 22 49 45 53 35 53 35 61 28 …
 $ NumberExistingCredits     : int  2 1 1 1 2 1 1 1 1 2 …
 $ NumberPeopleMaintenance   : int  1 1 2 2 2 2 1 1 1 1 …
 $ Telephone                 : num  0 1 1 1 1 0 1 0 1 1 …
 $ ForeignWorker             : num  1 1 1 1 1 1 1 1 1 1 …
 $ Class                     : Factor w/ 2 levels "Bad","Good": 2 1 2 2 1 2 2 2 2
1 …
```

task 1

[2]:
```
help(GermanCredit)
#It is a data set with 10 columns and 1000 rows, and has information about␣
 ↪People's Credit
```

task 2

[3]:
```
trainList <- createDataPartition(y=subCredit$Class,p=.40,list=FALSE)
```

task 3

[4]:
```
trainList[1:400] #I did it like this just so I could see everything so I could␣
 ↪make more sense of it
```

```
1. 1 2. 7 3. 12 4. 13 5. 15 6. 21 7. 22 8. 25 9. 28 10. 29 11. 31 12. 40 13. 45 14. 47 15. 50 16. 51
17. 54 18. 55 19. 57 20. 58 21. 59 22. 61 23. 62 24. 63 25. 64 26. 69 27. 73 28. 74 29. 75 30. 76 31. 77
32. 83 33. 89 34. 91 35. 100 36. 102 37. 103 38. 105 39. 106 40. 108 41. 113 42. 116 43. 117 44. 118
45. 121 46. 123 47. 125 48. 126 49. 128 50. 129 51. 131 52. 134 53. 135 54. 142 55. 143 56. 145
57. 150 58. 151 59. 159 60. 161 61. 166 62. 167 63. 168 64. 169 65. 170 66. 172 67. 173 68. 174
69. 183 70. 185 71. 186 72. 187 73. 191 74. 193 75. 195 76. 199 77. 205 78. 211 79. 214 80. 215
81. 217 82. 220 83. 222 84. 227 85. 229 86. 233 87. 234 88. 235 89. 239 90. 241 91. 243 92. 244
93. 246 94. 249 95. 253 96. 254 97. 260 98. 261 99. 262 100. 265 101. 269 102. 271 103. 273 104. 274
105. 276 106. 283 107. 288 108. 289 109. 291 110. 292 111. 293 112. 294 113. 300 114. 301 115. 306
116. 312 117. 313 118. 315 119. 316 120. 317 121. 321 122. 325 123. 327 124. 331 125. 336 126. 337
127. 338 128. 341 129. 347 130. 348 131. 349 132. 351 133. 353 134. 354 135. 363 136. 367 137. 370
138. 371 139. 373 140. 375 141. 377 142. 378 143. 380 144. 382 145. 383 146. 385 147. 386 148. 387
149. 393 150. 394 151. 395 152. 396 153. 399 154. 404 155. 405 156. 409 157. 410 158. 411 159. 416
160. 420 161. 422 162. 423 163. 427 164. 428 165. 432 166. 433 167. 434 168. 436 169. 439 170. 442
171. 448 172. 451 173. 452 174. 453 175. 455 176. 461 177. 462 178. 465 179. 467 180. 468 181. 469
182. 470 183. 473 184. 476 185. 480 186. 481 187. 486 188. 488 189. 489 190. 490 191. 491 192. 494
193. 498 194. 501 195. 503 196. 504 197. 505 198. 510 199. 513 200. 514 201. 519 202. 521 203. 523
204. 529 205. 531 206. 533 207. 537 208. 538 209. 542 210. 545 211. 546 212. 547 213. 548 214. 551
215. 553 216. 556 217. 559 218. 563 219. 565 220. 566 221. 568 222. 569 223. 570 224. 573 225. 576
226. 577 227. 579 228. 581 229. 582 230. 585 231. 589 232. 592 233. 593 234. 597 235. 598 236. 599
237. 600 238. 608 239. 611 240. 615 241. 617 242. 618 243. 619 244. 620 245. 621 246. 623 247. 624
248. 625 249. 626 250. 627 251. 629 252. 630 253. 631 254. 632 255. 634 256. 635 257. 636 258. 637
259. 641 260. 645 261. 652 262. 655 263. 659 264. 660 265. 662 266. 667 267. 669 268. 678 269. 680
270. 686 271. 690 272. 691 273. 693 274. 695 275. 698 276. 699 277. 701 278. 702 279. 706 280. 712
281. 714 282. 715 283. 720 284. 722 285. 725 286. 726 287. 733 288. 735 289. 736 290. 737 291. 741
292. 745 293. 746 294. 747 295. 750 296. 751 297. 754 298. 755 299. 757 300. 759 301. 760 302. 762
```

```
303. 767 304. 777 305. 781 306. 782 307. 783 308. 786 309. 787 310. 793 311. 798 312. 799 313. 801
314. 804 315. 811 316. 812 317. 814 318. 816 319. 817 320. 820 321. 822 322. 828 323. 829 324. 830
325. 831 326. 832 327. 835 328. 836 329. 839 330. 840 331. 844 332. 845 333. 847 334. 848 335. 850
336. 852 337. 854 338. 857 339. 858 340. 860 341. 866 342. 867 343. 868 344. 872 345. 873 346. 883
347. 886 348. 891 349. 895 350. 896 351. 897 352. 900 353. 901 354. 910 355. 911 356. 912 357. 916
358. 917 359. 918 360. 920 361. 921 362. 922 363. 923 364. 926 365. 927 366. 928 367. 930 368. 931
369. 934 370. 936 371. 940 372. 941 373. 943 374. 945 375. 950 376. 952 377. 953 378. 954 379. 955
380. 957 381. 958 382. 964 383. 965 384. 968 385. 970 386. 971 387. 973 388. 974 389. 975 390. 976
391. 977 392. 980 393. 985 394. 989 395. 992 396. 993 397. 994 398. 996 399. 997 400. 1000
```

task 4

```
[5]: #Trainlist is partitioning off 40% of the data for class to be used in the␣
      ↪training phase of supervised learning, and it is giving the indices of each␣
      ↪random selection it made, I don't really understand what you mean by␣
      ↪balanced for the next part of the question
```

task 5

```
[6]: trainset<-subCredit[trainList,]
     dim(trainset)
```

```
1. 400 2. 10
```

task 6

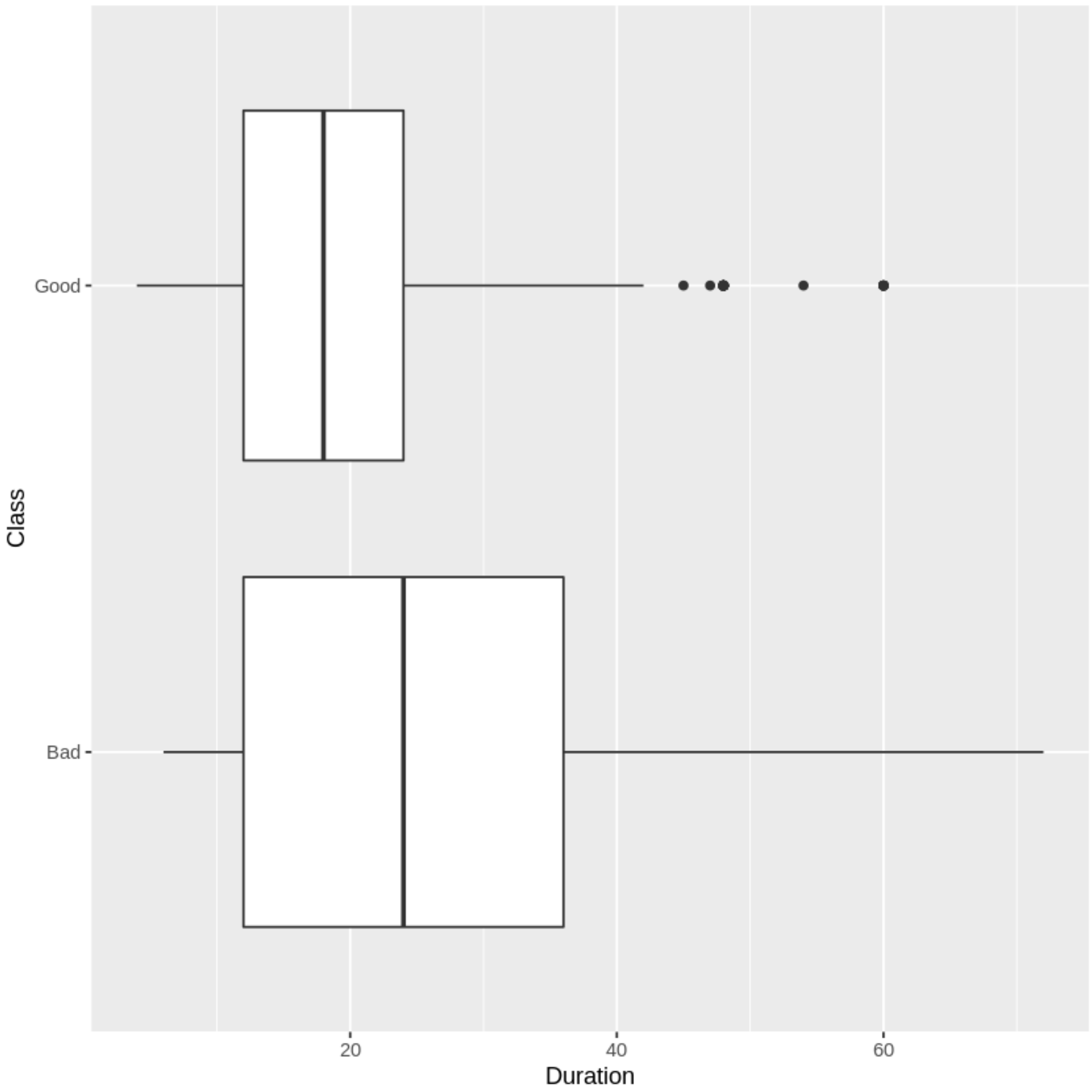
```
[7]: testset<-subCredit[-trainList,]
     dim(testset)
```

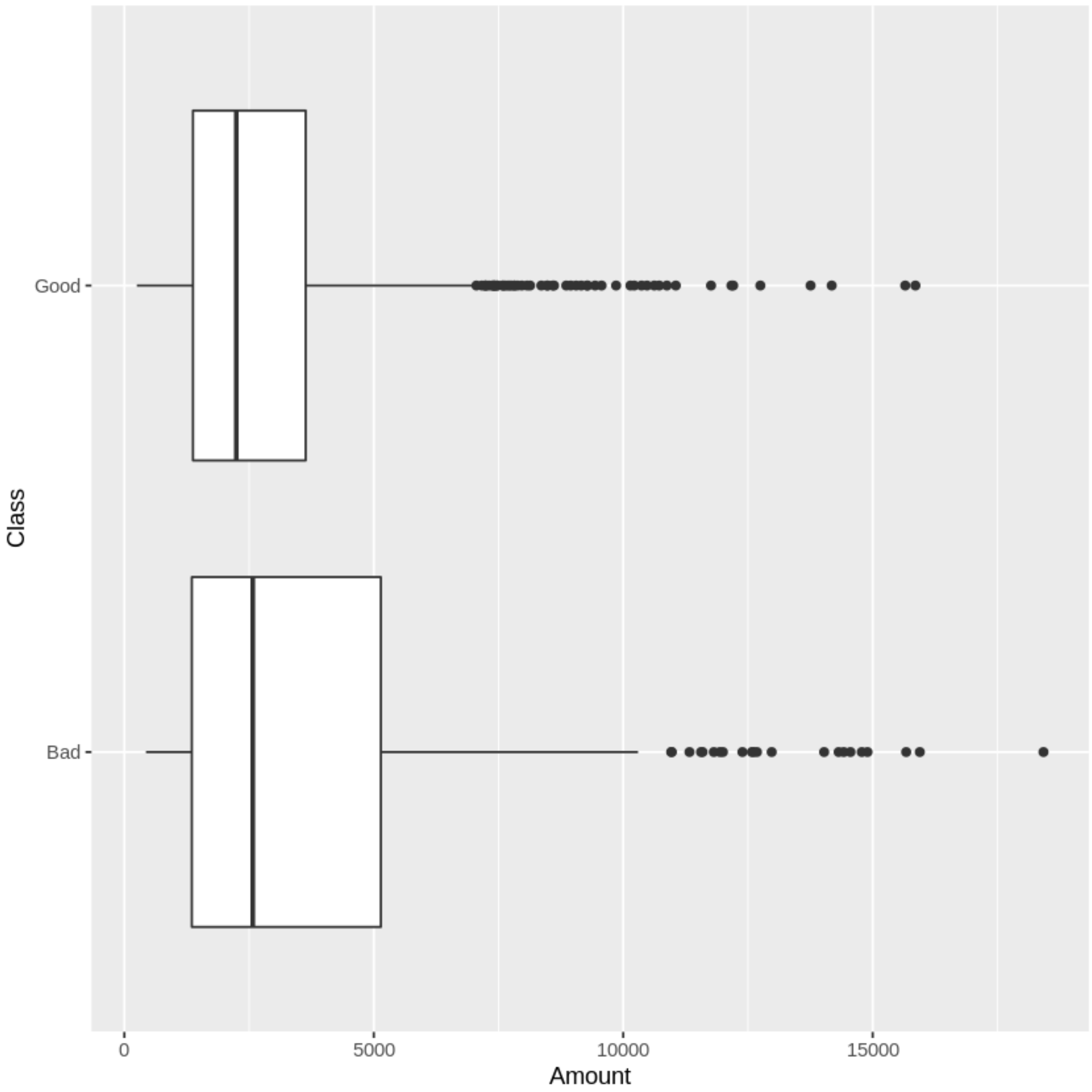
```
1. 600 2. 10
```

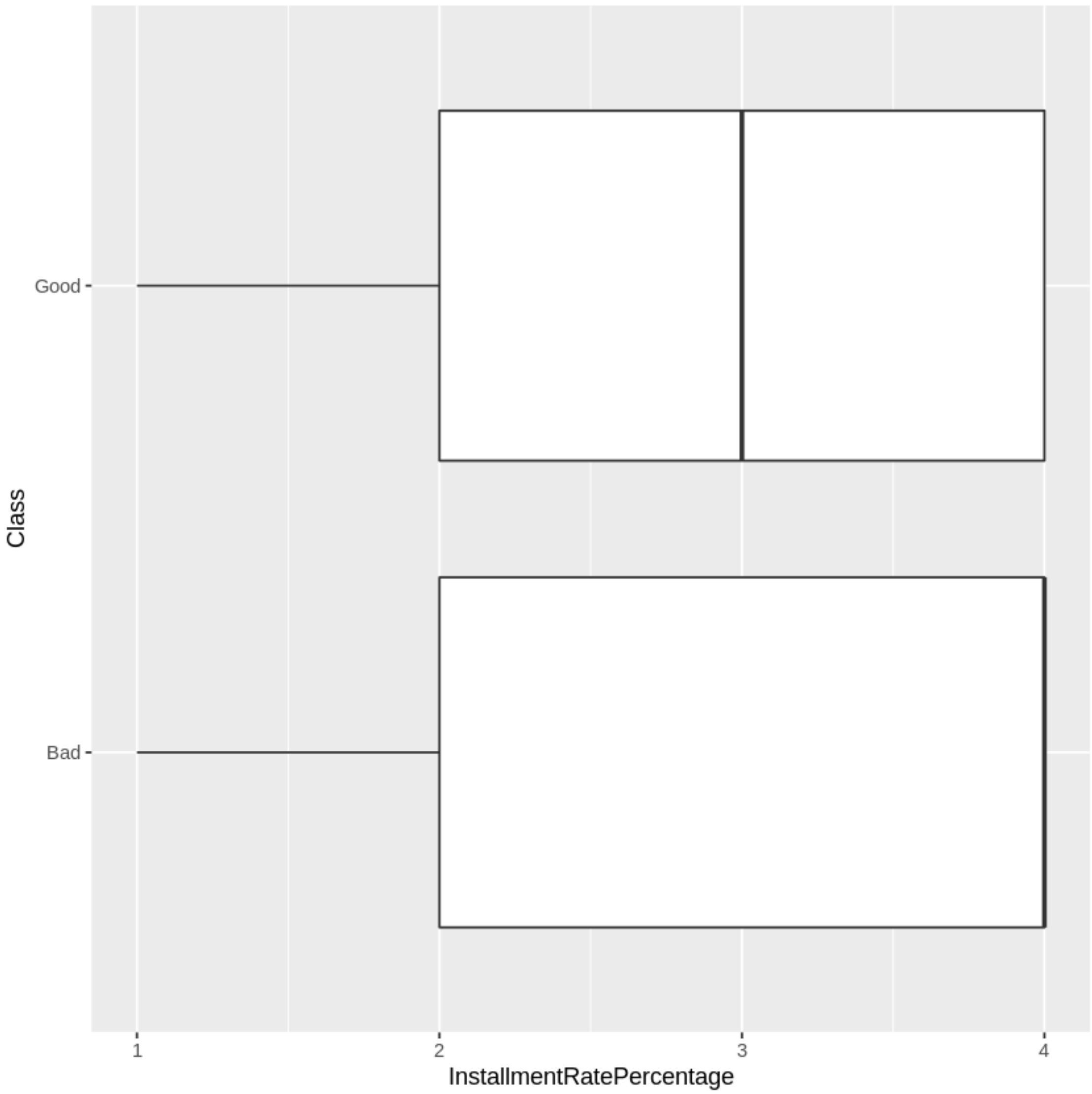
task 7

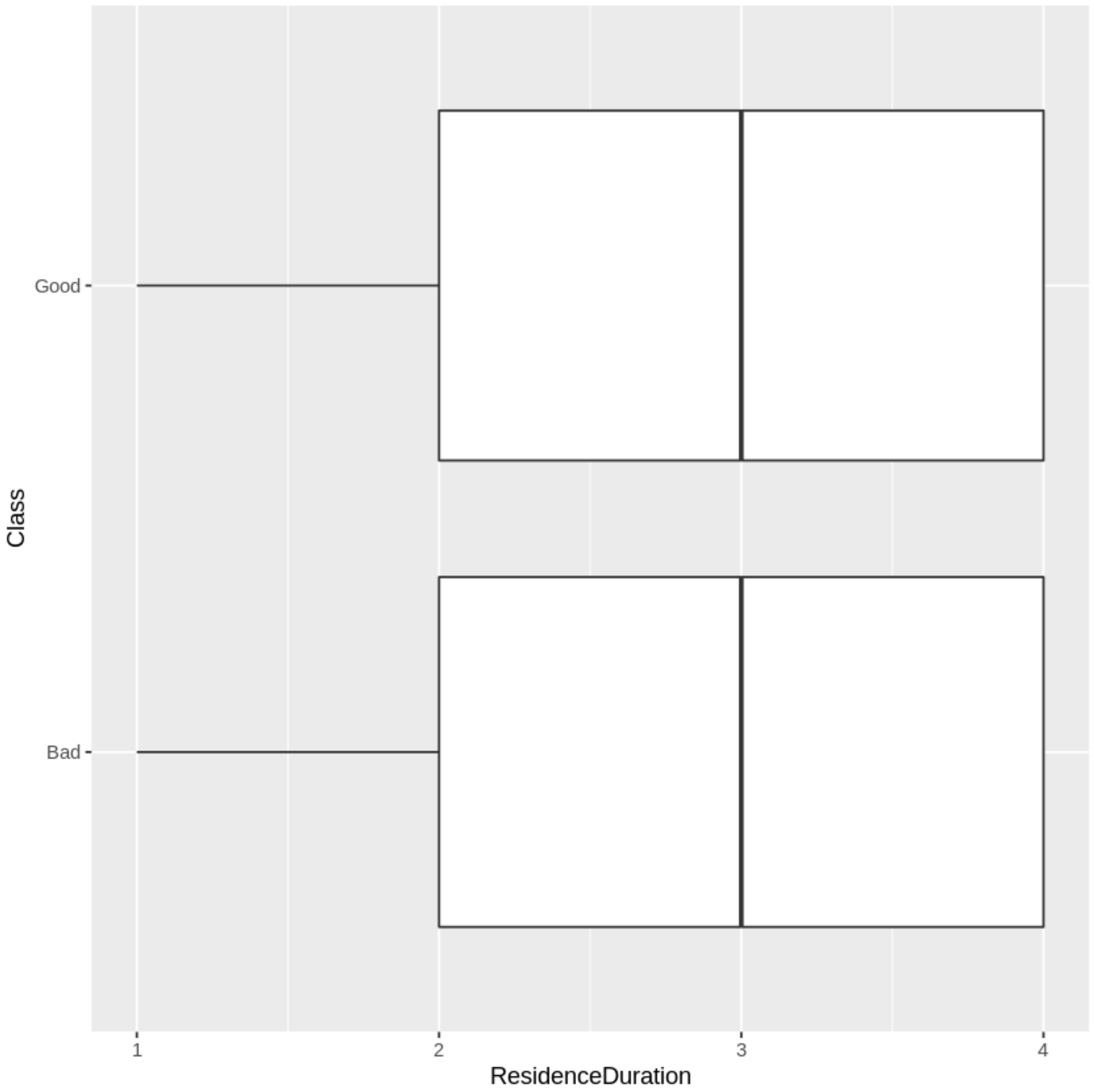
```
[8]: ggplot(subCredit,aes(x=Duration,y=Class))+geom_boxplot()
     #At shorter durations the class might go either way but as the duration␣
      ↪increases further it is more likely it will be bad
     ggplot(subCredit,aes(x=Amount,y=Class))+geom_boxplot()
     #Similar to duration at smaller amounts to a could go either way but at larger␣
      ↪amounts it's more likely to end up bad
     ggplot(subCredit,aes(x=InstallmentRatePercentage,y=Class))+geom_boxplot()
     #For installment rate percent the length of the bar on both are the same size␣
      ↪so it's equally likely regardless of what happens for it to go either way
     ggplot(subCredit,aes(x=ResidenceDuration,y=Class))+geom_boxplot()
     #Similar to the previous one the bars are the same length so regardless of how␣
      ↪long residence stay it could go either way
     ggplot(subCredit,aes(x=Age,y=Class))+geom_boxplot()
```

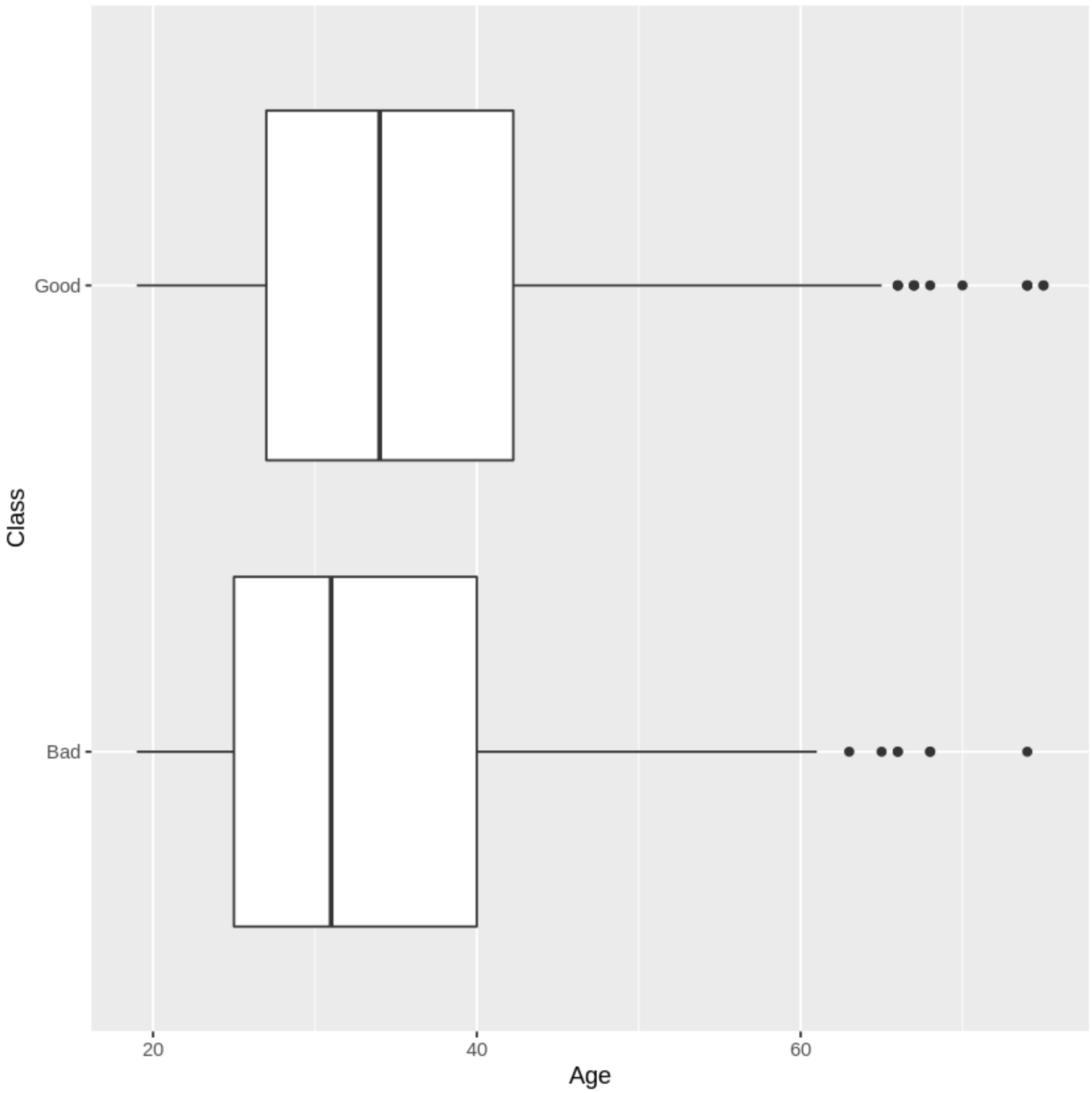
```
#4 age the bars are about the same size but one of them is slightly shifted to the left, people who are younger are more likely to go with bad and and people who are older are slightly more likely to go with good but a majority of both of the bars is in the same range and people in that range could go either way
ggplot(subCredit,aes(x=NumberExistingCredits,y=Class))+geom_boxplot()
#This one is just like residence duration and installment rate percentage in that the bars are the same size and cover the same range so it could go either way regardless
ggplot(subCredit,aes(x=NumberPeopleMaintenance,y=Class))+geom_boxplot()
#This one for some reason doesn't really give much of a graph and I don't know what to make of it, I think what is happening here is that it can be one or two and they vast majority of them are 1 creating this small sliver by both
ggplot(subCredit,aes(x=Telephone,y=Class))+geom_boxplot()
#Telephone is just the same as number of existing credits in that the bars are the same size and cover the same range so it could go either way
ggplot(subCredit,aes(x=ForeignWorker,y=Class))+geom_boxplot()
#This one is the exact same as number people maintenance, but flipped so I think the same thing could be happening here but with most of the values being 1 and the few values being 0 not 2 this time
```

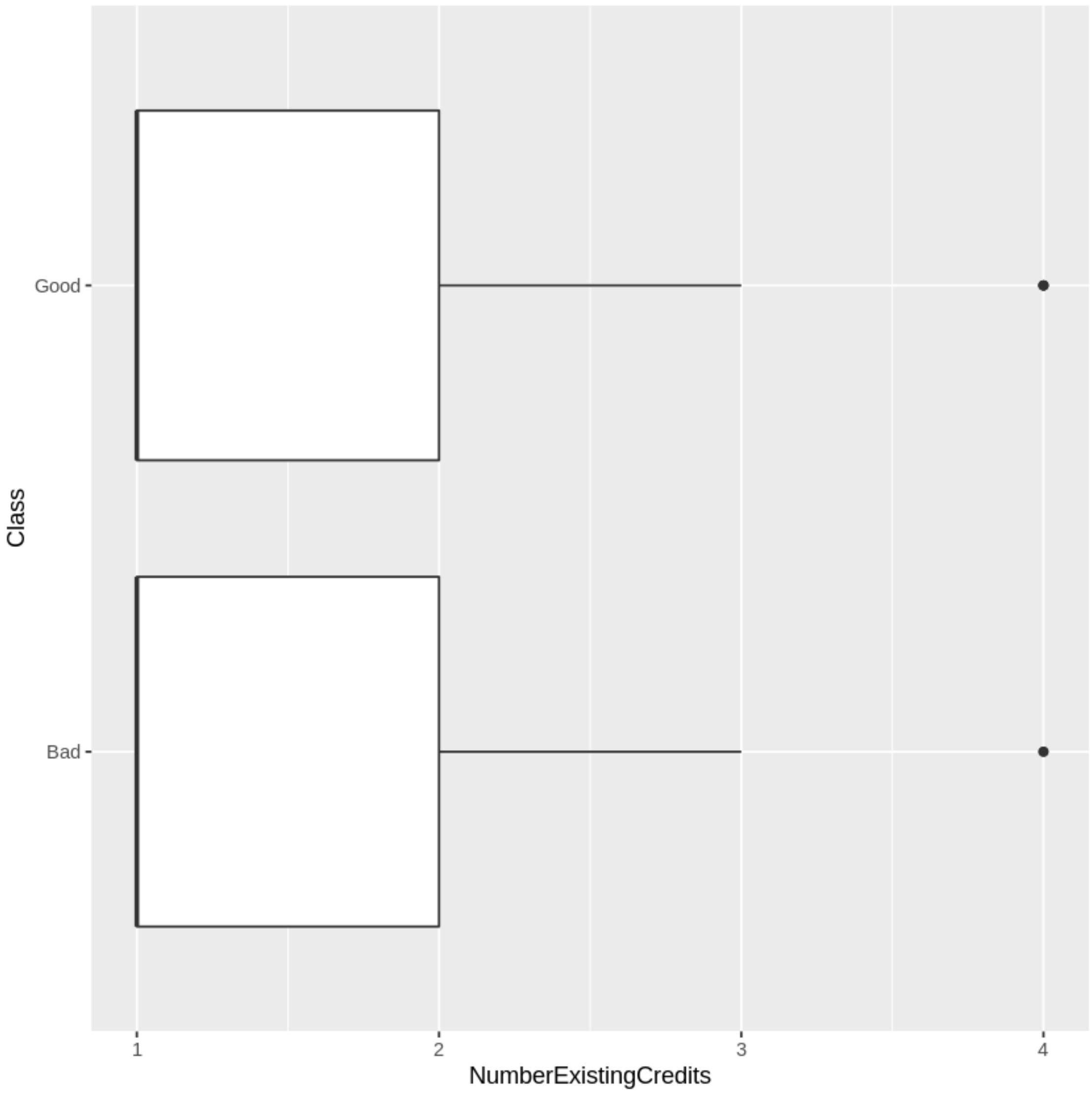

Good
Bad
Class
20
40
60
Duration

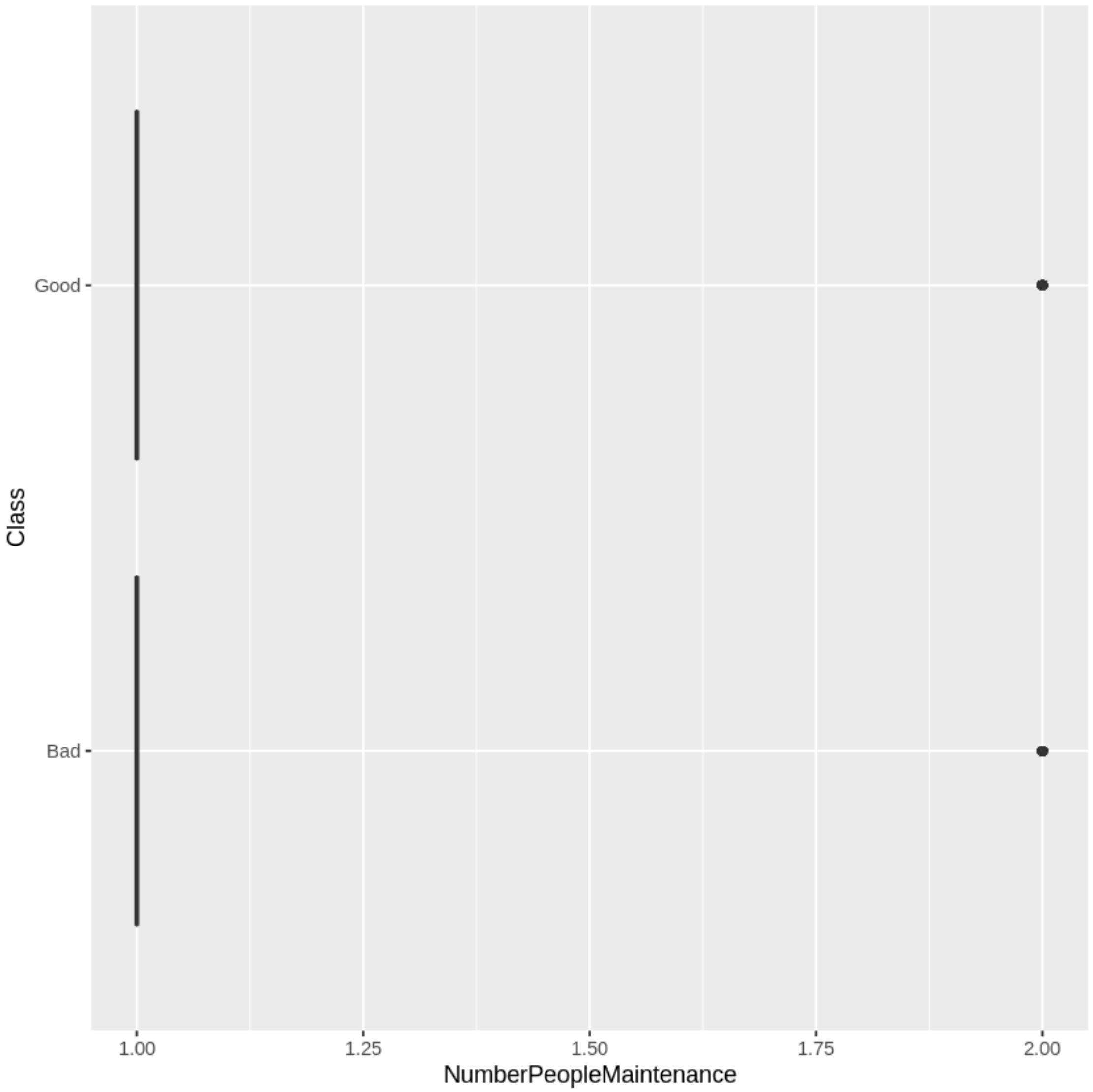

Good
Bad
Class
0
5000
10000
15000
Amount

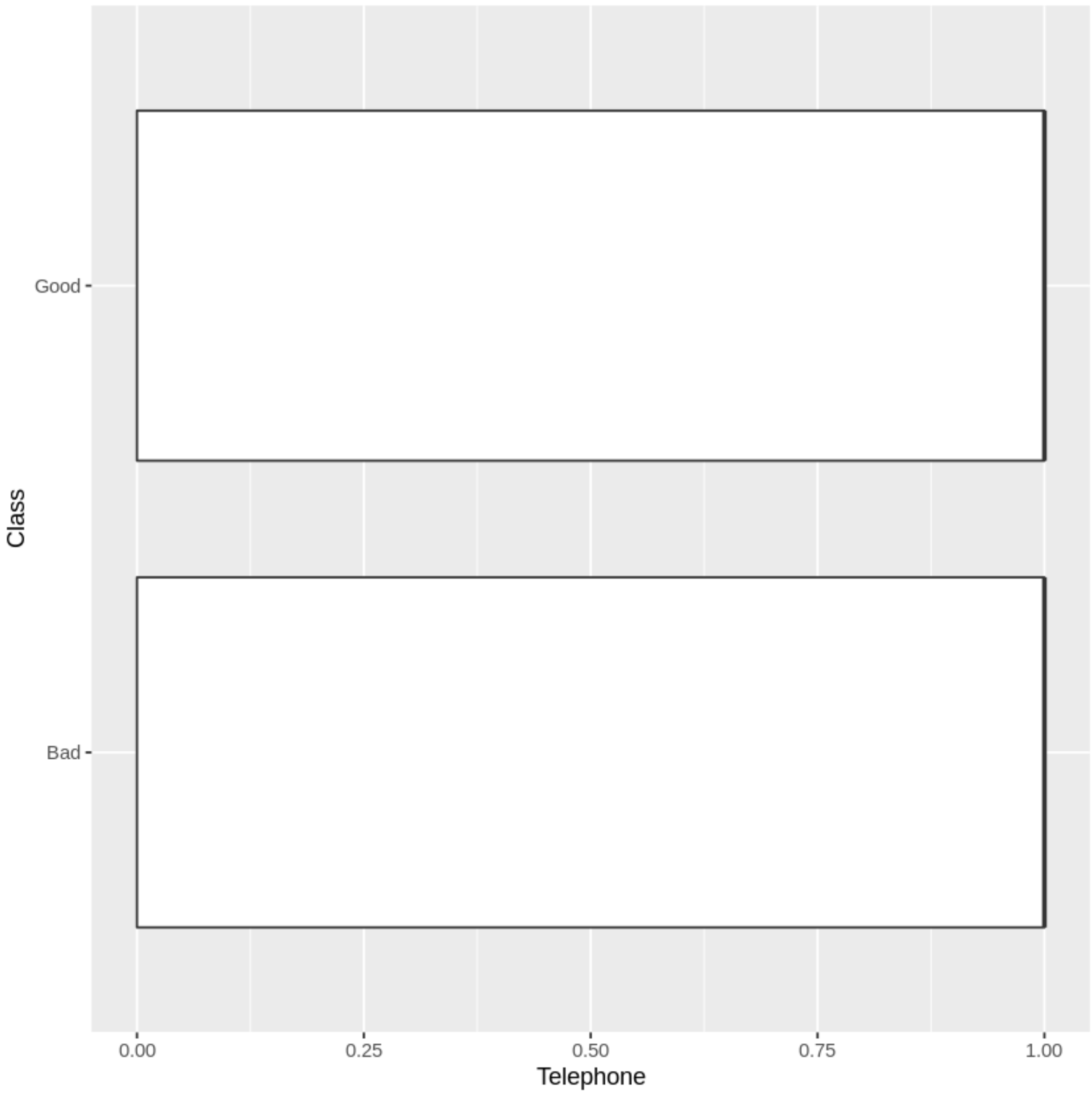

Good
Bad
Class
1
2
3
4
InstallmentRatePercentage

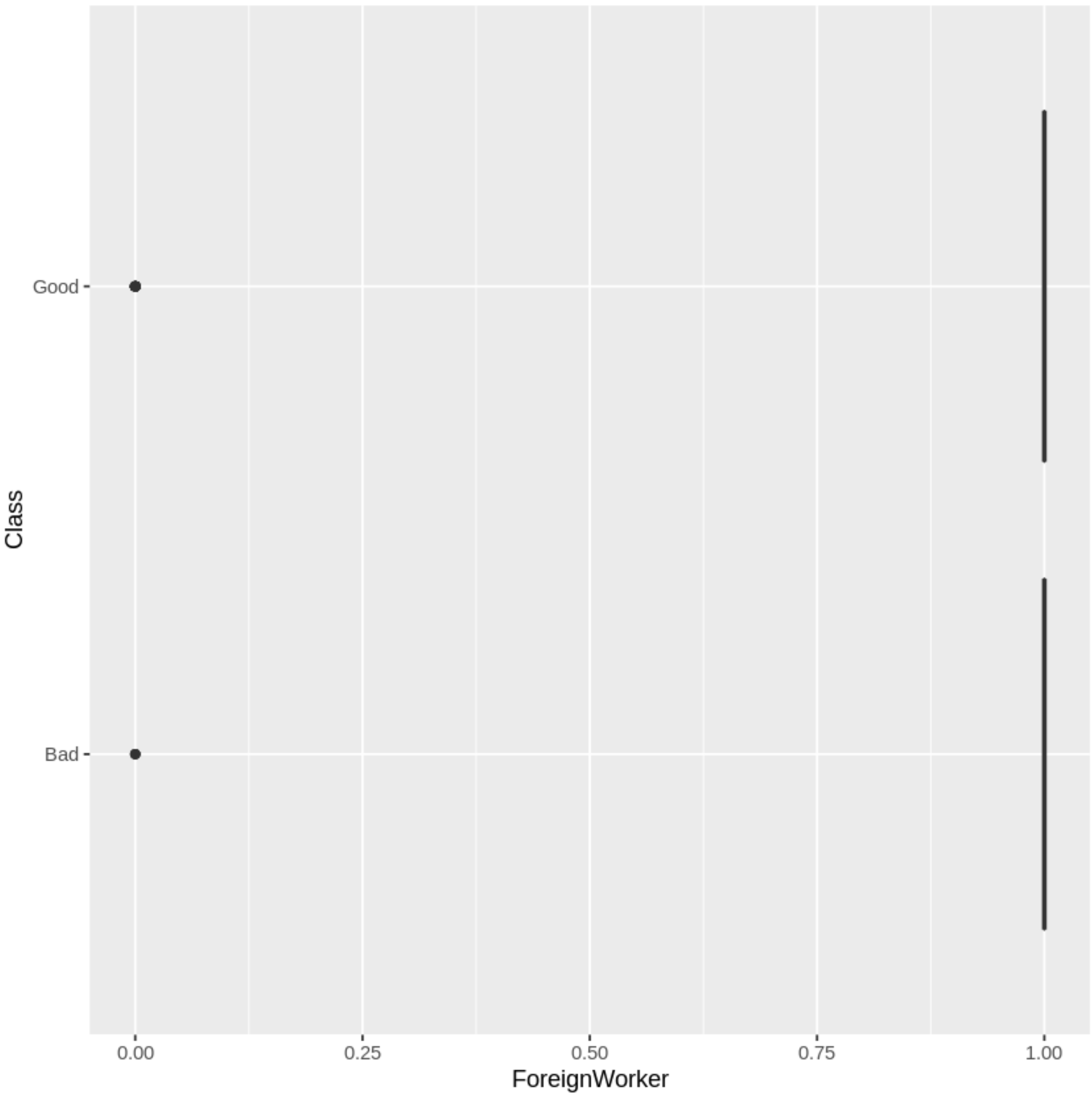

Good
Bad
Class
1
2
3
4
ResidenceDuration


Good
Bad
Class
20
40
60
Age


Good
Bad
Class
1
2
3
4
NumberExistingCredits


Good
Bad
Class
1.00
1.25
1.50
1.75
2.00
NumberPeopleMaintenance


Good
Bad
Class
0.00
0.25
0.50
0.75
1.00
Telephone



task 8

[9]:
```
ksvm<-ksvm(Class ~ ., data=trainset, kernel= "rbfdot", kpar = "automati",C = 5,␣
 ↪cross = 3, prob.model = TRUE)
```

task 9

[10]:
```
ksvm
#The cross validation error is at about 30% so it was an okay model not great␣
 ↪but not terrible
```

```
Support Vector Machine object of class "ksvm"

SV type: C-svc  (classification)
```

```
 parameter : cost C = 5

Gaussian Radial Basis kernel function.
 Hyperparameter : sigma =  0.108072099579765

Number of Support Vectors : 270

Objective Function Value : -918.0038
Training error : 0.19
Cross validation error : 0.312498
Probability model included.
```

task 10

[11]:
```
ksvmpred<- predict(ksvm,testset,type = "response")#votes was giving me a wierd␣
 ↪result so I took type from demo
conmatksvmpred<-data.frame(testset$Class,ksvmpred)
table(conmatksvmpred)
```

```
             ksvmpred
testset.Class Bad Good
         Bad   46  134
         Good  36  384
```

task 11

[12]:
```
str(ksvmpred)
```

```
 Factor w/ 2 levels "Bad","Good": 2 2 2 1 2 2 2 2 2 2 …
```

task 12

[15]:
```
#this model happens to be very sensitive, meaning it is very good at␣
 ↪determining when a result is a true positive, but it also comes with a lot␣
 ↪of false positives
sum(36,134)/600
#numbers have slighlty changed because I needed to rerun all cells
#our error is about 28.3% Which is a bit lower than what we got for our error␣
 ↪with the training data, but not by much
```

0.283333333333333

task 13

[16]:
```
#install.packages("e1071")
library(e1071)
confusionMatrix(testset$Class,ksvmpred)
#it defines positive class as bad I define it as good so it says the model is␣
 ↪specific and I say it's sensitive
#it is acurate about 71.7% of the time which means the error is about 28.3%␣
 ↪which is exactly what was calculated
```

```
also installing the dependency ‘proxy’


Updating HTML index of packages in '.Library'

Making 'packages.html' …
 done


Confusion Matrix and Statistics

          Reference
Prediction Bad Good
      Bad   46  134
      Good  36  384

               Accuracy : 0.7167
                 95% CI : (0.6788, 0.7524)
    No Information Rate : 0.8633
    P-Value [Acc > NIR] : 1

                  Kappa : 0.2011

 Mcnemar's Test P-Value : 1.01e-13

            Sensitivity : 0.56098
            Specificity : 0.74131
         Pos Pred Value : 0.25556
         Neg Pred Value : 0.91429
             Prevalence : 0.13667
         Detection Rate : 0.07667
   Detection Prevalence : 0.30000
      Balanced Accuracy : 0.65114

       'Positive' Class : Bad
```