Assignment 3

Sabbir Dewan

2024-10-23

#Question 1(a)

Read the dataset first

```
horse = read.csv("Horse.csv")
names(horse)
```

```
##  [1] "Age"                      "Sex"
##  [3] "Weight_carried"           "Place_finish"
##  [5] "Distance_from_winner"     "Race_speed"
##  [7] "Starts_this_prep"         "Days_in_this_racing_prep"
##  [9] "Track"                    "Race_distance"
## [11] "Number_starters"          "Temp"
## [13] "Outcome"
```

```
#check the horse dataset datatypes
str(horse)
```

```
## 'data.frame':    98 obs. of  13 variables:
##  $ Age                     : int  3 4 3 3 3 3 3 2 3 4 ...
##  $ Sex                     : chr  "F" "F" "F" "G" ...
##  $ Weight_carried          : num  55.5 54 54.5 57 56.5 57 57.5 55 55.5 56 ...
##  $ Place_finish            : int  4 12 6 6 2 3 1 1 6 8 ...
##  $ Distance_from_winner    : num  1.75 7.5 3.75 3.75 2.75 3.75 0 0 2 8.5 ...
##  $ Race_speed              : num  16.9 16.1 16.2 16 15.6 ...
##  $ Starts_this_prep        : int  1 1 1 1 1 1 4 1 2 8 ...
##  $ Days_in_this_racing_prep: int  0 0 0 0 0 0 31 0 10 160 ...
##  $ Track                   : int  2 1 0 1 0 2 2 0 0 1 ...
##  $ Race_distance           : int  1208 1000 1200 1200 1100 1216 1690 1000 1100 1650 ...
##  $ Number_starters         : int  12 12 9 10 10 12 14 12 10 12 ...
##  $ Temp                    : num  28.6 16.7 27.9 23.5 33.9 16.6 26.4 26.1 24.6 14.5 ...
##  $ Outcome                 : int  0 0 0 1 2 2 0 1 2 1 ...
```

Answer Question 1(a) -

The dataset consists of 98 observations and 13 variables, including 10 numerical (e.g., Age, Weight_carried, Race_speed) and 3 categorical variables (e.g., Sex, Track & Outcome). For the principal component analysis (PCA), only numerical variables were selected, as PCA requires numerical data for accurate dimensionality reduction. Categorical variables, such as Sex (Filly or Gelding) and Track (indicating race tracks 0, 1, or 2), were excluded, as they represent categorical behaviors that PCA cannot handle. Additionally, the Outcome variable, which reflects injury status and is categorical, was excluded. Remaining variables were used throughout the analysis.

Question 1(b) -

Check the null value for all columns. If get any NA, will remove that.

```
colSums(is.na(horse))
```

```
##                      Age                      Sex           Weight_carried
##                        0                        0                        0
##             Place_finish     Distance_from_winner              Race_speed
##                        0                        0                        0
##         Starts_this_prep Days_in_this_racing_prep                    Track
##                        0                        0                        0
##            Race_distance          Number_starters                     Temp
##                        0                        0                        0
##                  Outcome
##                        0
```

No NA values in any column.

Exclude the categorical variables from the dataset and find the co-variance between variables.

```
# Exclude the sex, Track and outcome column
cov.horse.variables <- cov(horse[ , !(names(horse) %in% c("Sex", "Track","Outcome"))])
cov.horse.variables
```

```
##                              Age Weight_carried Place_finish
## Age                   0.260572270     0.05612245   0.22354303
## Weight_carried        0.056122449     1.68864401  -0.48921734
## Place_finish          0.223543025    -0.48921734   7.01672628
## Distance_from_winner  0.115253524    -0.31964023   5.39539238
## Race_speed            0.003241111     0.03819430   0.01375868
## Starts_this_prep      0.272249106     0.36703135  -0.05217757
## Days_in_this_racing_prep  5.023458868     6.75199874   1.30601725
## Race_distance        47.472438460    40.70960446  61.13454660
## Number_starters       0.188407322     0.02519461   0.88333684
## Temp                 -0.535314538     0.52764044  -0.84983169
##                      Distance_from_winner   Race_speed Starts_this_prep
## Age                         0.115253524  0.003241111       0.27224911
## Weight_carried             -0.319640227  0.038194298       0.36703135
## Place_finish                5.395392384  0.013758679      -0.05217757
## Distance_from_winner        6.055322954 -0.002036188      -0.22604671
## Race_speed                 -0.002036188  0.151302935      -0.04443089
## Starts_this_prep           -0.226046707 -0.044430886       1.84052178
## Days_in_this_racing_prep   -2.081653692 -1.071294972      37.83925942
## Race_distance              37.812770882 -9.687905533     176.64148959
## Number_starters             0.351851462  0.192416369       0.15884704
## Temp                       -1.184119503  0.062434147      -0.43513570
##                      Days_in_this_racing_prep Race_distance Number_starters
## Age                                  5.023459     47.472438      0.18840732
## Weight_carried                       6.751999     40.709604      0.02519461
## Place_finish                         1.306017     61.134547      0.88333684
## Distance_from_winner                -2.081654     37.812771      0.35185146
## Race_speed                          -1.071295     -9.687906      0.19241637
```

```
## Starts_this_prep                        37.839259    176.641490      0.15884704
## Days_in_this_racing_prep               865.002840   3174.513676      4.54923206
## Race_distance                         3174.513676  49041.705975     59.48274774
## Number_starters                          4.549232     59.482748      4.69356196
## Temp                                   -10.586861    -42.063676     -3.57036608
##                                  Temp
## Age                        -0.53531454
## Weight_carried              0.52764044
## Place_finish               -0.84983169
## Distance_from_winner       -1.18411950
## Race_speed                  0.06243415
## Starts_this_prep           -0.43513570
## Days_in_this_racing_prep -10.58686093
## Race_distance            -42.06367557
## Number_starters           -3.57036608
## Temp                       32.58371660
```

All the covariances are non-zero, indicating some level of dependency between the variables. However, since covariance is sensitive to the scale of the variables, interpreting the strength of these relationships directly from the covariance values can be misleading. For example, Age ranges between 2 and 4 years, while Days_in_this_racing_prep ranges from 0 to 178 days, highlighting the different scales of the variables.

Let's do more clearer understanding of the relationship between variables, it's ideal to standardize them so that they are on the same scale. I will use standardize values for Principal Component Analysis. This allows for a more interpretable measure of dependence among the variables.

```r
# Calculate correlation excluding  Sex, Track and Outcome column
cor.horse.variables <- cor(horse[ , !(names(horse) %in% c("Sex", "Track", "Outcome"))])
cor.horse.variables
```

```
##                                  Age Weight_carried Place_finish
## Age                       1.00000000    0.084606448   0.16532166
## Weight_carried            0.08460645    1.000000000  -0.14212338
## Place_finish              0.16532166   -0.142123377   1.00000000
## Distance_from_winner      0.09175325   -0.099959419   0.82772698
## Race_speed                0.01632323    0.075562411   0.01335321
## Starts_this_prep          0.39312611    0.208191803  -0.01451931
## Days_in_this_racing_prep  0.33460312    0.176666633   0.01676382
## Race_distance             0.41994717    0.141463711   0.10421654
## Number_starters           0.17036579    0.008949265   0.15392452
## Temp                     -0.18371501    0.071132604  -0.05620376
##                          Distance_from_winner   Race_speed Starts_this_prep
## Age                             0.091753251  0.016323231       0.39312611
## Weight_carried                 -0.099959419  0.075562411       0.20819180
## Place_finish                    0.827726981  0.013353213      -0.01451931
## Distance_from_winner            1.000000000 -0.002127284      -0.06771097
## Race_speed                     -0.002127284  1.000000000      -0.08419588
## Starts_this_prep               -0.067710972 -0.084195875       1.00000000
## Days_in_this_racing_prep       -0.028762793 -0.093643281       0.94833850
## Race_distance                   0.069388430 -0.112466566       0.58794873
## Number_starters                 0.065999315  0.228332147       0.05404523
## Temp                           -0.084299805  0.028118891      -0.05618934
##                          Days_in_this_racing_prep Race_distance Number_starters
## Age                                   0.33460312    0.41994717     0.170365793
```

```
## Weight_carried                          0.17666663    0.14146371     0.008949265
## Place_finish                            0.01676382    0.10421654     0.153924516
## Distance_from_winner                   -0.02876279    0.06938843     0.065999315
## Race_speed                             -0.09364328   -0.11246657     0.228332147
## Starts_this_prep                        0.94833850    0.58794873     0.054045231
## Days_in_this_racing_prep                1.00000000    0.48740014     0.071396685
## Race_distance                           0.48740014    1.00000000     0.123981521
## Number_starters                         0.07139668    0.12398152     1.000000000
## Temp                                   -0.06306060   -0.03327546    -0.288709557
##                                    Temp
## Age                          -0.18371501
## Weight_carried                0.07113260
## Place_finish                 -0.05620376
## Distance_from_winner         -0.08429980
## Race_speed                    0.02811889
## Starts_this_prep             -0.05618934
## Days_in_this_racing_prep     -0.06306060
## Race_distance                -0.03327546
## Number_starters              -0.28870956
## Temp                          1.00000000
```

Now,will run the Principal component analysis on the reduced horse dataset. I use prcomp function and there Scale =TRUE parameter will ensure the scaling of all variables.

```
horse_reduced <-  horse[, !names(horse) %in% c("Sex", "Track", "Outcome")]
horse.pca <- prcomp(horse_reduced , scale = TRUE)
horse.pca
```

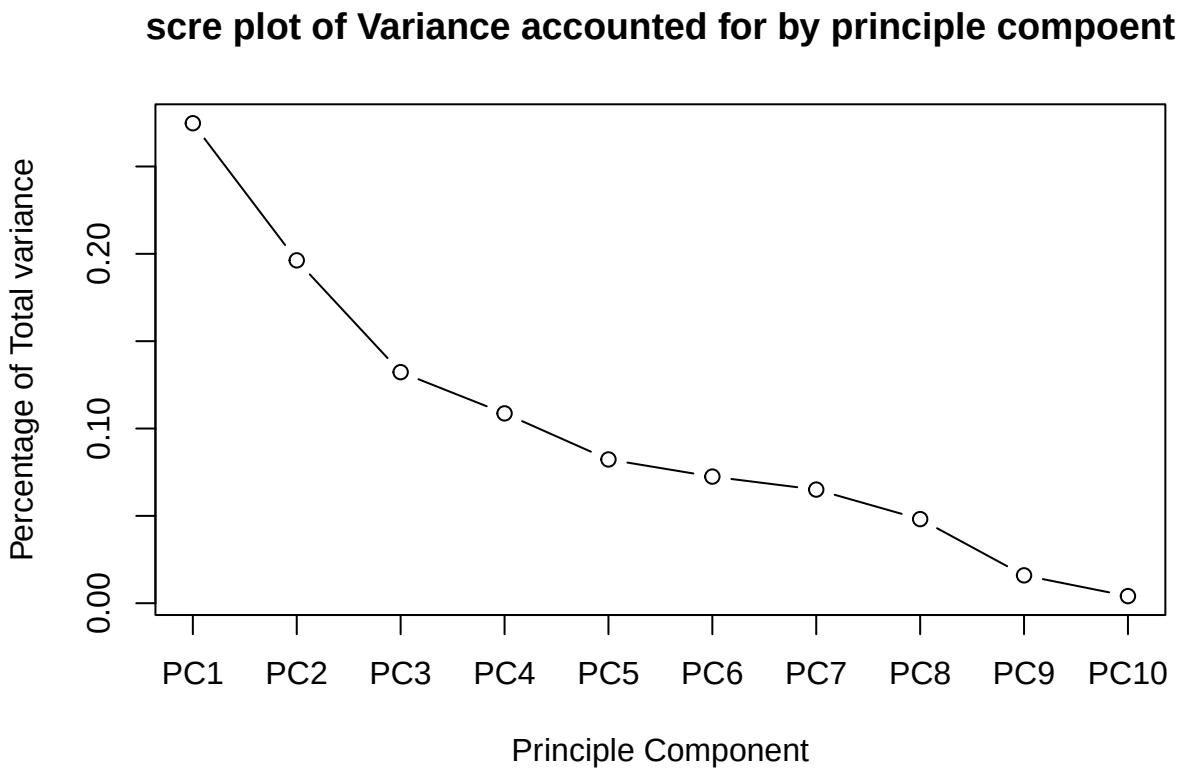
```
## Standard deviations (1, .., p=10):
##  [1] 1.6575990 1.4009594 1.1500975 1.0425048 0.9072023 0.8513379 0.8065523
##  [8] 0.6939585 0.3994042 0.2017838
##
## Rotation (n x k) = (10 x 10):
##                                    PC1          PC2         PC3          PC4
## Age                          -0.37683316  0.124167176 -0.14878461 -0.057561667
## Weight_carried               -0.15791156 -0.211028480 -0.08873835  0.540512241
## Place_finish                 -0.08907967  0.644521671  0.19548628  0.149360663
## Distance_from_winner         -0.05113167  0.633535599  0.23527204  0.152602086
## Race_speed                    0.05824904  0.064327614 -0.50857729  0.579012048
## Starts_this_prep             -0.55116327 -0.151803594  0.09017707  0.014345367
## Days_in_this_racing_prep     -0.52712844 -0.126798843  0.09751092  0.001206738
## Race_distance                -0.45798401  0.008525194  0.08252745  0.007799207
## Number_starters              -0.12831007  0.218676864 -0.62893242 -0.022273593
## Temp                          0.10611296 -0.179224396  0.44470543  0.568258769
##                                    PC5         PC6          PC7         PC8
## Age                          -0.01733220 -0.77526259 -0.227198454 -0.39856635
## Weight_carried               -0.78979257  0.04608734  0.017093305 -0.03436311
## Place_finish                 -0.01729653  0.07725566  0.007864517 -0.08173484
## Distance_from_winner         -0.12608654  0.12495300 -0.086882313  0.05947654
## Race_speed                    0.40622331  0.01282781 -0.406588876  0.26005738
## Starts_this_prep              0.15466907  0.27300434 -0.121203758 -0.08746443
## Days_in_this_racing_prep      0.16781512  0.39723649 -0.145863563 -0.21747448
## Race_distance                 0.04798032 -0.28171121  0.365084628  0.74568681
```

```
## Number_starters           0.05136748  0.16915056  0.655169381 -0.26429575
## Temp                      0.37106302 -0.18653768  0.420313629 -0.28507880
##                                  PC9         PC10
## Age                       0.066590670  0.029799159
## Weight_carried           -0.054049586  0.014568436
## Place_finish             -0.709299768  0.004114871
## Distance_from_winner      0.687691606 -0.043140021
## Race_speed               -0.009981051  0.018365449
## Starts_this_prep         -0.010426440 -0.737518844
## Days_in_this_racing_prep  0.044576279  0.663348108
## Race_distance            -0.023453066  0.108431523
## Number_starters           0.091505990 -0.030689027
## Temp                      0.074159619 -0.004529682
```

This result is the square root of eigenvalues and eigenvectors, on other words the standard deviation and rotation.
Eigenvalues represent the total variance captured by a PCA. On the other hand, Eigenvector values represent the contribution of each variable to the principal components which means rotating the data into new dimensions(Orthogonal) to achieve uncorrelated variables.

I will plot a Scree plot to find optimal Principal component.

```
plot(horse.pca$sdev^2/sum(horse.pca$sdev^2), xaxt = "n", xlab = "Principle Component", ylab = "Percentag
axis(side = 1, at = 1:10, labels = c("PC1", "PC2", "PC3", "PC4", "PC5", "PC6", "PC7", "PC8", "PC9", "PC
```



Add a more reader firnedly scree plot from ggplot.

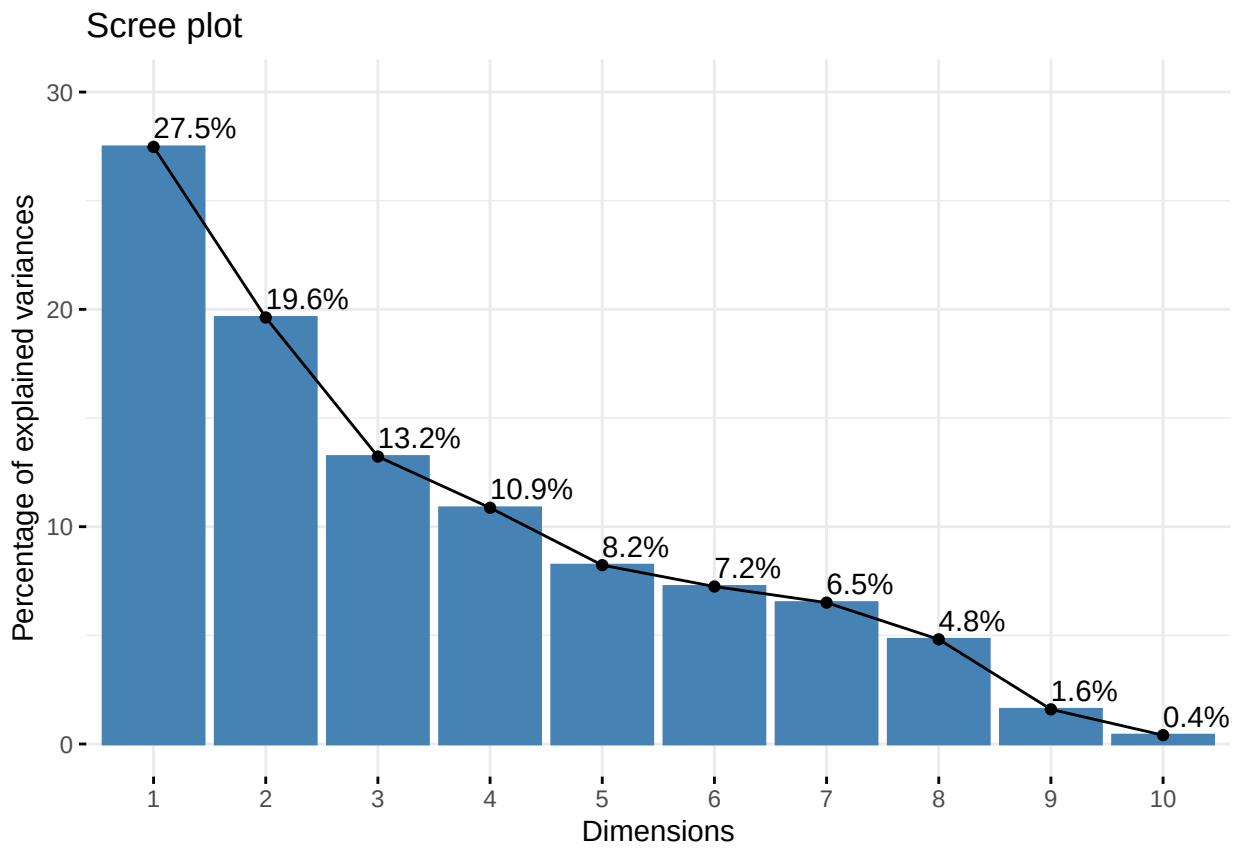
```r
library(MASS)
library(factoextra)
```

```
## Warning: package 'factoextra' was built under R version 4.3.3
```

```
## Loading required package: ggplot2
```

```
## Welcome! Want to learn more? See two factoextra-related books at https://goo.gl/ve3WBa
```

```r
library(ggplot2)

fviz_eig(horse.pca, addlabels = TRUE, ylim = c(0,30))
```




```r
#Summary of principle component analysis
summary(horse.pca)
```

```
## Importance of components:
##                           PC1    PC2    PC3    PC4    PC5     PC6     PC7
## Standard deviation     1.6576 1.4010 1.1501 1.0425 0.9072 0.85134 0.80655
## Proportion of Variance 0.2748 0.1963 0.1323 0.1087 0.0823 0.07248 0.06505
## Cumulative Proportion  0.2748 0.4710 0.6033 0.7120 0.7943 0.86677 0.93182
##                           PC8     PC9    PC10
## Standard deviation     0.69396 0.39940 0.20178
## Proportion of Variance 0.04816 0.01595 0.00407
## Cumulative Proportion  0.97998 0.99593 1.00000
```

Ans 1(b) -

As per Elbow method from Scree plot - few principle components contribute highest in terms of explain variability. First four PCA explain 71% of variability and after this four PCA, contribution of PCA drastically getting lower which means not contributing for the variability. So, ideally choosing 4 PCA as per elbow method.

Ans 1(b) -

Considering 70% of total variation, first 4 PCA is enough as we can see from the Cumulative Proportion which is 71% upto 4 PCA.

Qsn 1(b) -

To get the PC using 1 cut off let check the eigenvalues first.

Get the eigenvalues (variance explained by each principal component)

```
# Get the eigenvalues (variance explained by each principal component)
eigenvalues <- (horse.pca$sdev)^2
print(eigenvalues)
```

```
##  [1] 2.74763440 1.96268730 1.32272435 1.08681631 0.82301606 0.72477626
##  [7] 0.65052657 0.48157833 0.15952372 0.04071669
```

```
components_above_1 <- which(eigenvalues >= 1)
components_above_1
```

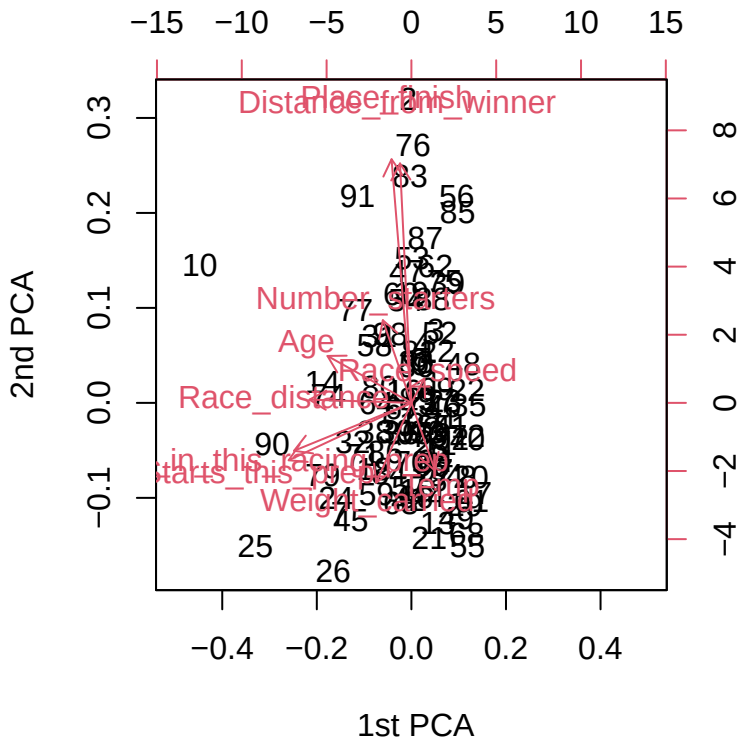
```
## [1] 1 2 3 4
```

#Answer 1(b) - As we consider 1 as a cut-off, we will select 4 PCA.

#Question 1(c) -

Produce a biplot of first 2 PCA

```
biplot(horse.pca, xlim = c(-0.5, 0.5), xlab= " 1st PCA", ylab = "2nd PCA")
```




```
horse_reduced_2 <-  horse[, !names(horse) %in% c("Sex", "Track", "Outcome", "Days_in_this_racing_prep")]
horse.pca_2 <- prcomp(horse_reduced_2 , scale = TRUE)
horse.pca_2
```

```
## Standard deviations (1, .., p=9):
## [1] 1.4701400 1.3652855 1.1425908 1.0425040 0.9009431 0.8135750 0.7485053
## [8] 0.6241893 0.3985980
##
## Rotation (n x k) = (9 x 9):
##                                PC1         PC2         PC3          PC4
## Age                    -0.47205028  0.17101789  0.06356487 -0.056641475
## Weight_carried         -0.08075178  0.31544923  0.03033690  0.541183094
## Place_finish           -0.37766273 -0.54003815 -0.16885139  0.149286126
## Distance_from_winner   -0.33400328 -0.55521095 -0.20448198  0.152534139
## Race_speed              0.01530954 -0.07228467  0.53748444  0.578224783
## Starts_this_prep       -0.41674903  0.40868723 -0.14534735  0.014669142
## Race_distance          -0.48055995  0.29963595 -0.17886278  0.008884525
## Number_starters        -0.25904617 -0.07318931  0.62485523 -0.022733246
## Temp                    0.21095838  0.06289322 -0.43847939  0.568511014
##                                PC5         PC6           PC7          PC8
## Age                     0.18274039  0.57447982  0.6098054698  0.03977539
## Weight_carried         -0.76268140  0.02188066  0.1127755441 -0.04568231
## Place_finish           -0.03979064 -0.03604008  0.0002218409  0.11724037
## Distance_from_winner   -0.16043530  0.02860480 -0.1148210558 -0.01758524
## Race_speed              0.36161812  0.31251835 -0.3656037121 -0.09526076
```

```
## Starts_this_prep      0.10894276 -0.08198817 -0.4248124219  0.66014574
## Race_distance         0.18405534 -0.26782810 -0.1888276419 -0.71004822
## Number_starters       0.02042860 -0.63405424  0.3252987989  0.13907581
## Temp                  0.42505338 -0.29871492  0.3829206008  0.11789370
##                              PC9
## Age                   0.06169892
## Weight_carried       -0.05660911
## Place_finish         -0.70590720
## Distance_from_winner  0.68924033
## Race_speed           -0.01144788
## Starts_this_prep      0.05572592
## Race_distance        -0.04215433
## Number_starters       0.09473148
## Temp                  0.07512634
```

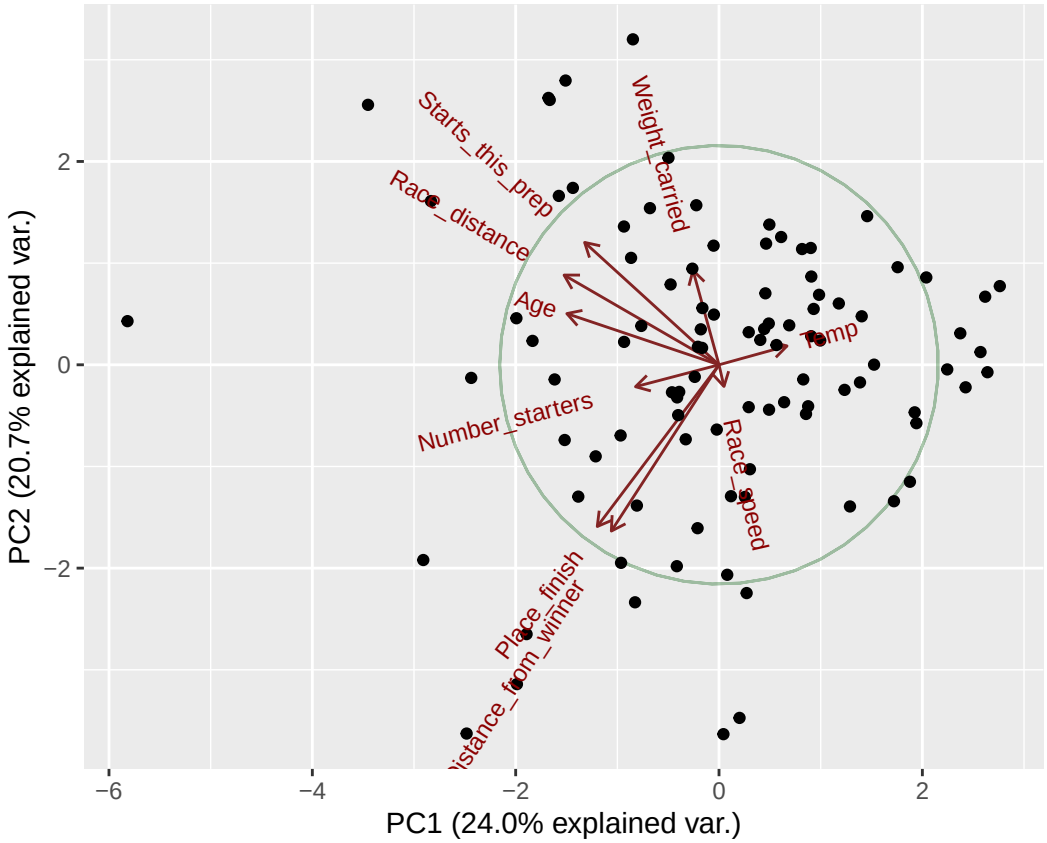
```
library(ggbiplot)
```

```
## Loading required package: plyr
```

```
## Loading required package: scales
```

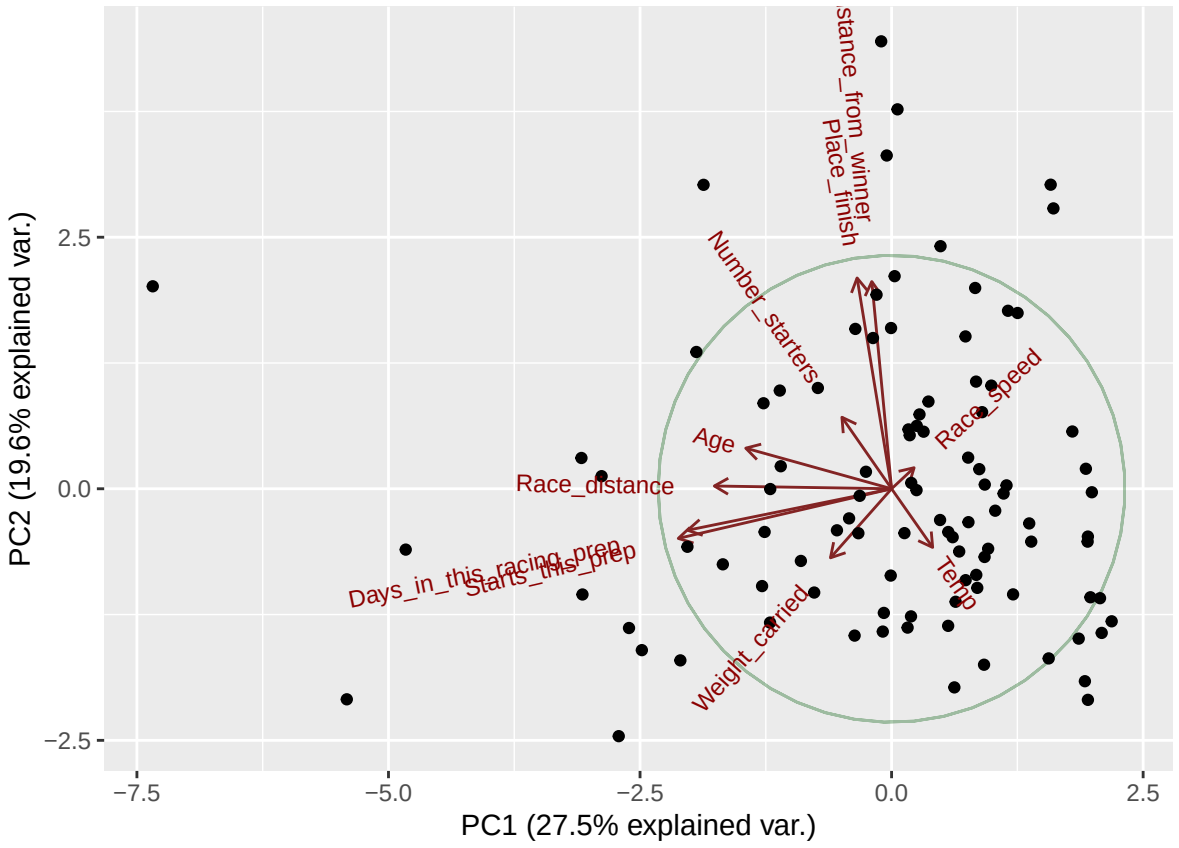
```
## Loading required package: grid
```

```
# Create the biplot
g <- ggbiplot(horse.pca_2,
              obs.scale = 1, var.scale = 1,
               ellipse = TRUE, circle = TRUE)
g <- g + scale_color_discrete(name = '')
g <- g + theme(legend.direction = 'horizontal', legend.position = 'top')

print(g)
```



Print the more reader friend biplot from ggbiplot package

```
library(ggbiplot)
# Create the biplot
g <- ggbiplot(horse.pca,
              obs.scale = 1, var.scale = 1,
               ellipse = TRUE, circle = TRUE)

g <- g + scale_color_discrete(name = '')

g <- g + theme(legend.direction = 'horizontal', legend.position = 'top')

print(g)
```



get the loading of PCA1 & PCA2

```
# Get the loading for first 2 PCA
loadings <- horse.pca$rotation[, 1:2]
loadings
```

```
##                                PC1          PC2
## Age                     -0.37683316  0.124167176
## Weight_carried          -0.15791156 -0.211028480
## Place_finish            -0.08907967  0.644521671
## Distance_from_winner    -0.05113167  0.633535599
## Race_speed               0.05824904  0.064327614
## Starts_this_prep        -0.55116327 -0.151803594
## Days_in_this_racing_prep -0.52712844 -0.126798843
## Race_distance           -0.45798401  0.008525194
## Number_starters         -0.12831007  0.218676864
## Temp                     0.10611296 -0.179224396
```

PC1 = -0.377 × Age - 0.158 × Weight_carried - 0.089 × Place_finish - 0.051 × Distance_from_winner + 0.059 × Race_speed - 0.551 × Starts_this_prep - 0.527 × Days_in_this_racing_prep - 0.458 × Race_distance - 0.128 × Number_starters + 0.106 × Temp

PC2 = 0.124 × Age - 0.211 × Weight_carried + 0.644 × Place_finish + 0.634 × Distance_from_winner + 0.064 × Race_speed - 0.152 × Starts_this_prep - 0.127 × Days_in_this_racing_prep + 0.008 × Race_distance + 0.219 × Number_starters - 0.179 × Temp

#Ans 1(c)

The vectors for Days_in_this_racing_prep, Starts_this_prep, and Race_distance extend further from the origin in the horizontal direction (corresponding to the first principal component) compared to other variables, indicating their higher loading on the first principal component.

On the other hand, the vectors for Place_finish and Distance_from_winner extend further from the origin in the vertical direction (corresponding to the second principal component) than other variables, consistent with their far greater loading on the second principal component.

#Qsn 1(d)

Check the variance percentage

```
loadings_pc1 <- horse.pca_2$rotation[, 1:1]

# Square the loading for PC1
squared_loadings_pc1 <- loadings_pc1^2

# Calculate the total sum of squared loading for PC1
total_squared_pc1 <- sum(squared_loadings_pc1)

# Calculate percentage contributions
percentage_contributions_pc1 <- (squared_loadings_pc1 / total_squared_pc1) * 100

print(percentage_contributions_pc1)
```

```
##                  Age       Weight_carried         Place_finish
##          22.28314715           0.65208496          14.26291351
## Distance_from_winner           Race_speed     Starts_this_prep
##          11.15581890           0.02343821          17.36797567
##        Race_distance      Number_starters                 Temp
##          23.09378622           6.71049169           4.45034369
```

#Ans 1(d)-

The variable Starts_this_prep explains the largest proportion of variability, contributing approximately 30.4%, followed closely by Days_in_this_racing_prep at 27.8%. Age also makes a notable contribution of 14.2%. Race_distance contributes 21.0%, while Weight_carried has a moderate contribution of 2.49%. Number_starters and Temp each contribute 1.65% and 1.13%, respectively. Variables such as Place_finish and Distance_from_winner contribute more modestly, at 0.79% and 0.26%, respectively, while Race_speed contributes the least at 0.34%.

#Qsn2 (a) - Assumption of LDA

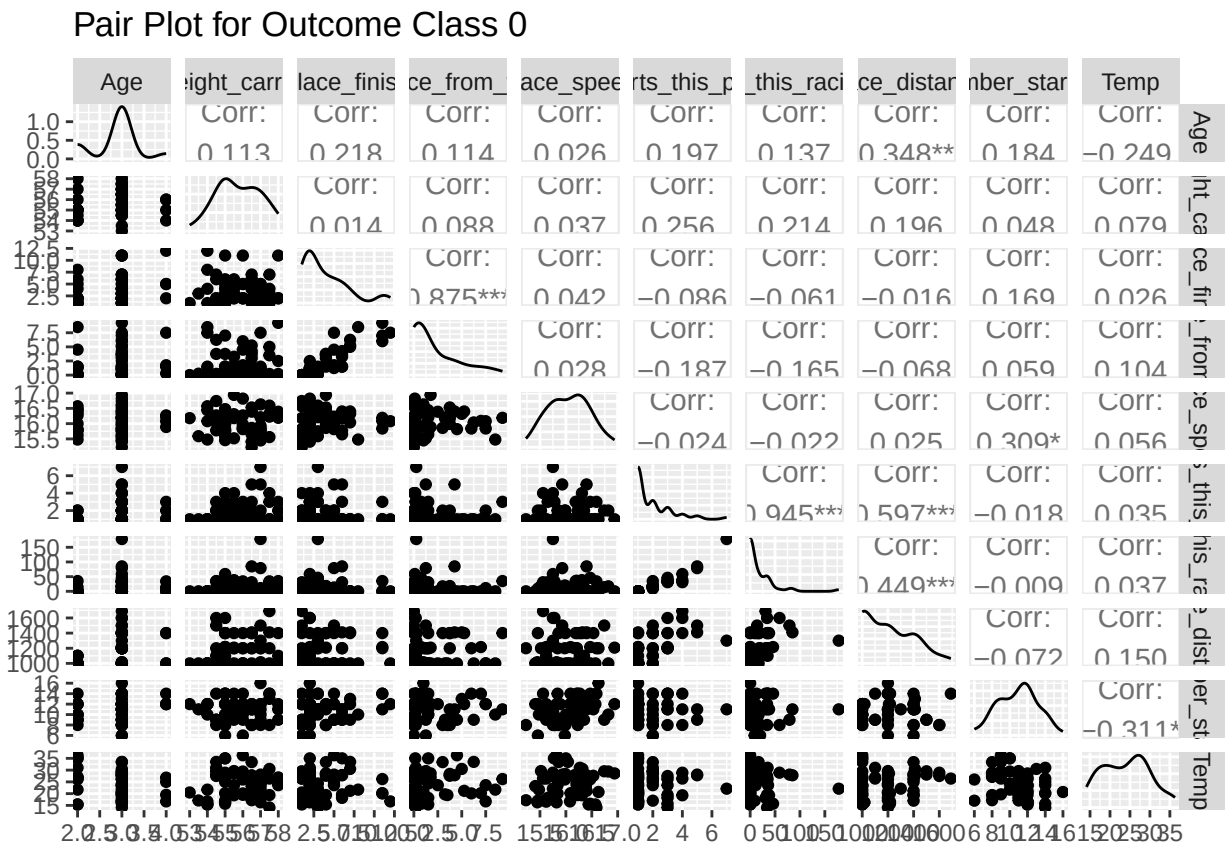
```
library(GGally)
```

```
## Warning: package 'GGally' was built under R version 4.3.3
```

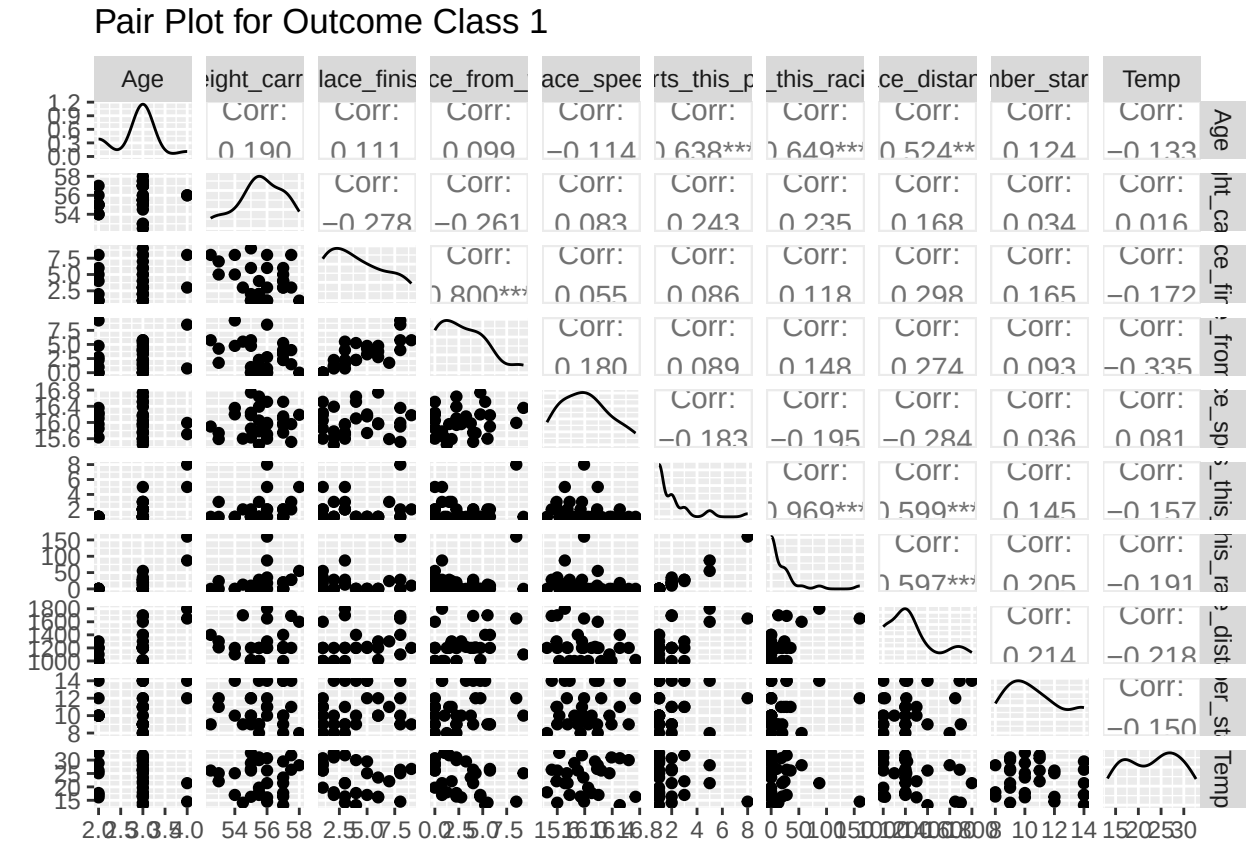
```
## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2
```

```
# Subset the data for each class of Outcome, excluding 'Sex', 'Track', and 'Outcome'
horse_class_0 <- subset(horse, Outcome == "0")[, !(names(horse) %in% c('Sex', 'Track', 'Outcome'))]
horse_class_1 <- subset(horse, Outcome == "1")[, !(names(horse) %in% c('Sex', 'Track', 'Outcome'))]
```

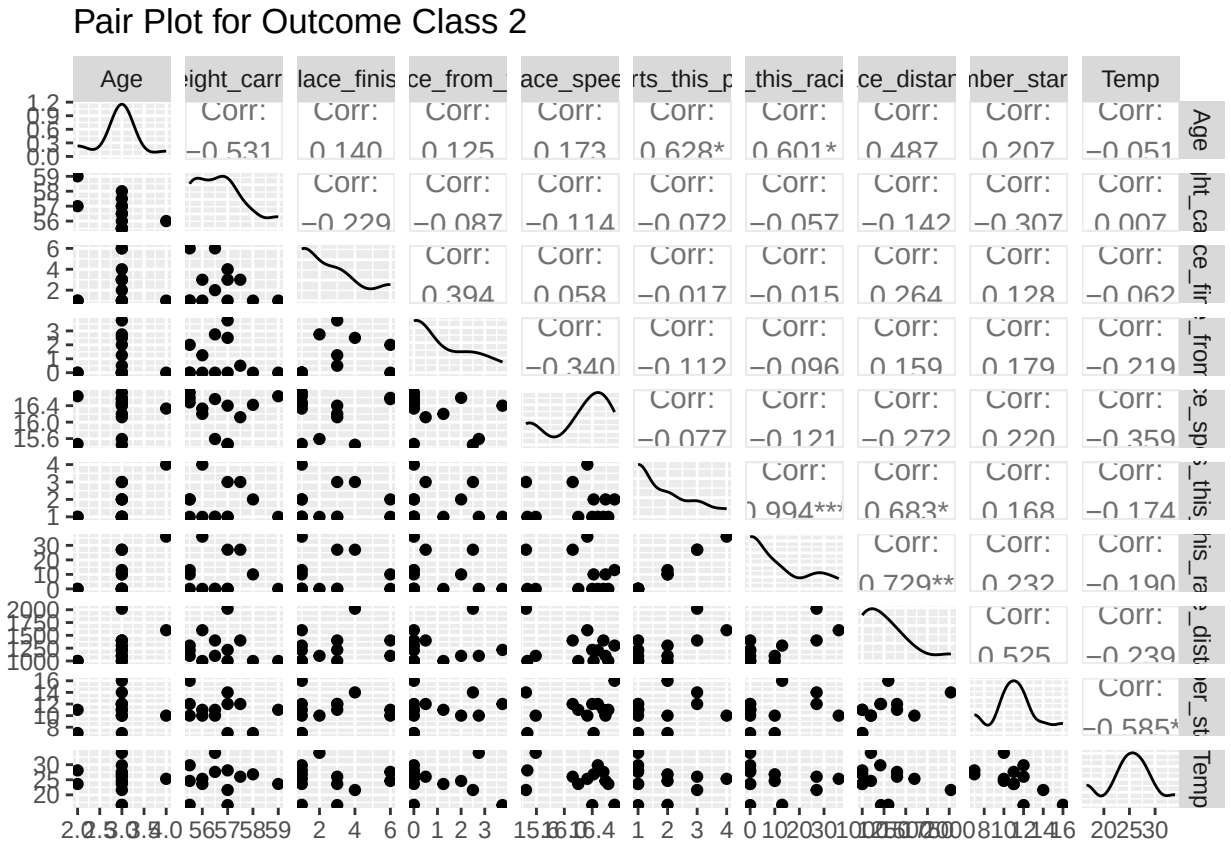
```
horse_class_2 <- subset(horse, Outcome == "2")[, !(names(horse) %in% c('Sex', 'Track', 'Outcome'))]

# Create pair plots for each class
ggpairs(horse_class_0,  progress = FALSE) + ggtitle("Pair Plot for Outcome Class 0")
```




```
ggpairs(horse_class_1, progress = FALSE) + ggtitle("Pair Plot for Outcome Class 1")
```




```
ggpairs(horse_class_2, progress = FALSE) + ggtitle("Pair Plot for Outcome Class 2")
```



This distributions for Outcome level 0 indicate Race_speed, Race_distance, and Number_starters are appear close to normal distribution, but others like Days_in_this_racing_prep and Age show significant skewness or multimodality.Also its clear that the relationships between some variables show linear patterns, while others (like Days_in_this_racing_prep vs. Race_speed) show a less clear, nonlinear relationship.

This distributions for Outcome level 1 indicate that Age, Days_in_this_racing_prep and Place_finish has multimodal,Right Skewed distribution indicating a violation of the normality assumption. Heteroscedasticity is observed in relationships like Race_distance vs. Days_in_this_racing_prep, where variance increases with the values of one variable. Furthermore, strong correlations between some variables, such as Days_in_this_racing_prep and Starts_this_prep, suggest that the assumption of independence may not hold.

This distributions for Outcome level 2 indicate that Age and Weight_carried with Weight_carried showing multimodal,Right Skewed distribution. Heteroscedasticity is also present, particularly in the relationship between Starts_this_prep and Race_distance. Additionally, strong correlations, such as between Days_in_this_racing_prep and Starts_this_prep (0.994) suggest a violation of independence assumptions.

#Qsn 2(b) - I will not use the Days_in_this_racing_prep this variable as this variable have a high co-relation (0.94) with Starts_this_prep column.

Let's apply the LDA rest of the numeric columns.

```
# Subset the data excluding 'Sex', 'Track', and 'Days_in_this_racing_prep' columns
horse_reduced_4 <- horse[ , !(names(horse) %in% c("Sex", "Track","Days_in_this_racing_prep"))]


lda_model_3 <- lda(Outcome ~ ., data = horse_reduced_4)
print(lda_model_3)
```

```
## Call:
## lda(Outcome ~ ., data = horse_reduced_4)
##
## Prior probabilities of groups:
##         0         1         2
## 0.5612245 0.3061224 0.1326531
##
## Group means:
##        Age Weight_carried Place_finish Distance_from_winner Race_speed
## 0 2.872727       55.76364     3.854545            2.1081818   16.05673
## 1 2.833333       55.61667     4.066667            2.9166667   16.00700
## 2 2.923077       56.69231     2.538462            0.9807692   16.22769
##   Starts_this_prep Race_distance Number_starters     Temp
## 0         1.872727      1196.327        10.89091 24.06545
## 1         1.933333      1246.000        10.76667 23.09000
## 2         1.769231      1257.154        11.00000 24.93846
##
## Coefficients of linear discriminants:
##                                LD1          LD2
## Age                    0.4729067600  0.483592782
## Weight_carried         0.4913704915 -0.135679677
## Place_finish           0.0778486672  0.426374727
## Distance_from_winner -0.3246742754 -0.457460517
## Race_speed             0.9909157915 -0.434271485
## Starts_this_prep     -0.3421902801  0.267331692
## Race_distance          0.0009846346 -0.003958869
## Number_starters        0.0105363065  0.053846026
## Temp                   0.0316048116  0.041325998
##
## Proportion of trace:
##    LD1    LD2
## 0.7736 0.2264
```

```r
# Predict using the LDA model
lda_predictions_3 <- predict(lda_model_3)
```

let's calculate the Hit Rate

```r
actual_classes <- horse_reduced_4$Outcome

predicted_classes <- lda_predictions_3$class

# Calculate hit rate
hit_rate <- mean(predicted_classes == actual_classes)

print(paste("Hit Rate:", hit_rate))
```

```
## [1] "Hit Rate: 0.612244897959184"
```

Now, I will check the hit rate without excluding the Days_in_this_racing_prep (highly co-related) column

```r
horse_reduced_3 <-  horse[, !names(horse) %in% c("Sex", "Track")]

lda_model_2 <- lda(Outcome ~ ., data = horse_reduced_3)
print(lda_model_2)
```

```
## Call:
## lda(Outcome ~ ., data = horse_reduced_3)
##
## Prior probabilities of groups:
##         0         1         2
## 0.5612245 0.3061224 0.1326531
##
## Group means:
##        Age Weight_carried Place_finish Distance_from_winner Race_speed
## 0 2.872727       55.76364     3.854545            2.1081818   16.05673
## 1 2.833333       55.61667     4.066667            2.9166667   16.00700
## 2 2.923077       56.69231     2.538462            0.9807692   16.22769
##   Starts_this_prep Days_in_this_racing_prep Race_distance Number_starters
## 0         1.872727                16.472727      1196.327        10.89091
## 1         1.933333                17.533333      1246.000        10.76667
## 2         1.769231                 9.461538      1257.154        11.00000
##       Temp
## 0 24.06545
## 1 23.09000
## 2 24.93846
##
## Coefficients of linear discriminants:
##                                    LD1          LD2
## Age                         0.4074615324  0.537173710
## Weight_carried              0.4780752628 -0.102752994
## Place_finish                0.0699463788  0.422488894
## Distance_from_winner       -0.2966000979 -0.477485888
## Race_speed                  0.9280044432 -0.335247110
## Starts_this_prep            0.2389329792 -0.200574698
## Days_in_this_racing_prep -0.0260030282  0.019995402
## Race_distance               0.0006495928 -0.003544117
## Number_starters             0.0228657205  0.043133691
## Temp                        0.0304832980  0.042133307
##
## Proportion of trace:
##    LD1    LD2
## 0.7769 0.2231
```

```r
# Predict using the LDA model
lda_predictions_2 <- predict(lda_model_2)
```

Let's check the Hit Rate

```r
actual_classes <- horse_reduced_3$Outcome

# Predicted Outcome values
predicted_classes_2 <- lda_predictions_2$class
```

```r
# Calculate hit rate
hit_rate_2 <- mean(predicted_classes_2 == actual_classes)

print(paste("Hit Rate:", hit_rate_2))
```

```
## [1] "Hit Rate: 0.612244897959184"
```

Print the confusion matrix

```r
confusion.matrix <- table(lda_predictions_2$class, horse_reduced_3$Outcome)
print(confusion.matrix)
```

```
##
##      0  1  2
##   0 51 22 10
##   1  3  7  1
##   2  1  1  2
```

#Question 2(b) - Ans Excluding the variable "Days_in_this_racing_prep" due to its high correlation with Starts_this_preparation variable, doesn't improved the hit rate. Consequently, both model hit rate stands at 61%.

#Question 2(c) - Ans

Variables such as age, Weight_carried, Race_speed, Number_of_race_starts, and Number_of_starters showing almost same among all the 3 outcome levels. These variables do not significantly differ between horses Outcome Classes that experience no injuries, minor injuries, or moderate to severe injuries.

Horses that experience moderate to severe injuries (Outcome 2) tend to perform significantly better in races, with an average 2.53 in Place_finish and a distance from the winner is lowest 0.98 comparing to other 2 outcome. In comparison, horses experiencing no injuries (Outcome 0) finish with place finish of 3.85 and a distance of 2.10 from the winner, indicating they finish far behind. Horses with minor injuries (Outcome 1) have a place finish of 4.06 and a distance of 2.91.

Average higher temperate can cause the Horses with moderate to severe injuries (Outcome 2) with 24.93. On the other hand, horses with minor injuries (Outcome 1) race in lowest temperature (23.09) compared to other two outcome groups. Horses with no injuries (Outcome 0) race on average in the middle temperature (24.06) are faced by other 2 group.

Horses with injuries, especially those with minor or moderate/severe injuries, participate in slightly longer races like race distances is 1246 and 1257 respectively. Horses with no injuries (Outcome 0) race shorter distances, averaging a 1196.

#Question 2(d) -

```r
priors <- lda_model_2$prior
costs <- c(1, 20, 100)

# Initialize vector to store new priors
new_priors <- numeric(length(priors))
for (i in 1:length(priors)) {
  new_priors[i] <- (priors[i] * costs[i]) / sum(priors * costs)
}
names(new_priors) <- c(0, 1, 2)

new_priors
```

```
##          0          1          2
## 0.02813299 0.30690537 0.66496164
```

#Question 2(d) -i (Ans)

New Priors are for Outcome 0 is 3%, for Outcome 1 is 30% and for Outcome 2 is 67%.

Calculate New hit rate and misclassification rate

```
# Fit LDA with the new priors
lda_model_3 <- lda(Outcome ~ ., data = horse_reduced_3, prior = new_priors)
lda_predictions_4 <- predict(lda_model_3)

# Confusion matrix to evaluate performance
confusion.matrix.2 <- table(lda_predictions_4$class, horse_reduced_3$Outcome)
print(confusion.matrix.2)
```

```
##
##      0  1  2
##   0  0  0  0
##   1 26 16  0
##   2 29 14 13
```

The hit Rate of Outcome 2 increased to (13/13) 100% comparing to previous without any cost which was (2/13) 15%. The Hit rate of Outcome 1 also increased (16/30) 53% comparing to previous (7/30) 23%. But decreased drastically in Outcome 1 as it's hit rate now 0% (0/55) comparing to previous (51/55) 92%.

```
# Calculate the hit rate
correct_predictions <- sum(diag(confusion.matrix.2))
total_predictions <- sum(confusion.matrix.2)
hit_rate <- correct_predictions / total_predictions

misclassification_rate <- 1 - hit_rate

print(hit_rate)
```

```
## [1] 0.2959184
```

```
print(misclassification_rate)
```

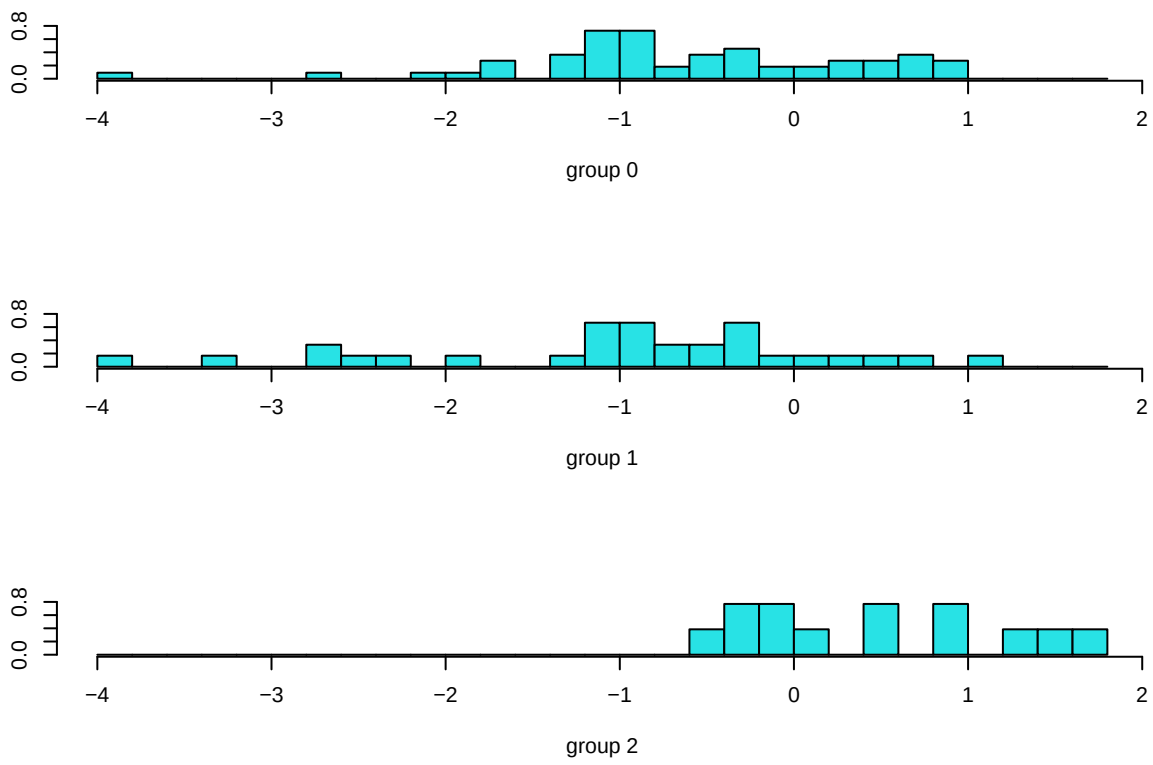
```
## [1] 0.7040816
```

#Ans 2(d) -ii New Hit rate is 30% and Misclassification rate is 70%

#Qsn 2(e) -

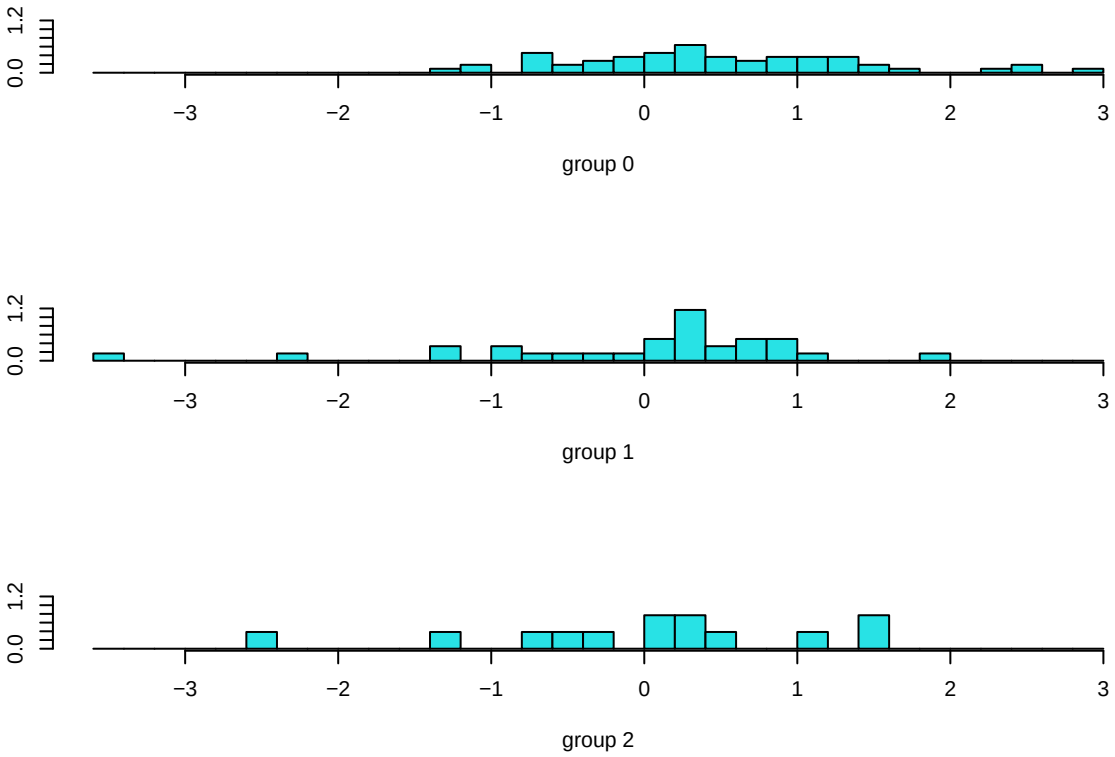
Plot the LDA histogram to check how well it's separated (feasibility of LDA)

```
lda_scores <- lda_predictions_4$x
#for the first LDA
ldahist(data = lda_scores[, 1], g = horse_reduced_3$Outcome)
```



Let's check the second LDA

```
lda_scores <- lda_predictions_4$x
#for the second LDA
ldahist(data = lda_scores[, 2], g = horse_reduced_3$Outcome)
```



Both LDA1 and LDA2 can't separate more accurately the outcome of horse injury as observations are overlapped between groups.

#Qsn 2(e) - ans

LDA is not effective in this context for predicting injury outcomes. The high misclassification rate, significant overlap in the LDA components, and poor separation between the injury categories suggest that LDA is not suitable for this problem.

#Question 3(a) - Ans -

Principal Components Regression (PCR) is suitable for predictive modeling when there are highly correlated predictor variables, which can lead to multicollinearity and inaccuracy in standard regression. PCR addresses these issues by transforming the predictors into principal components. Since each principal component is orthogonal to the previous ones, Principal Components Regression (PCR) is used to avoid errors caused by dependencies between the assumed independent variables in regression (Hadi and Ling, 1998). This orthogonality ensures that the transformed components are independent, thereby addressing multicollinearity issues.

Also, PCR particularly useful for high-dimensional datasets where the number of predictors is large or when predictors are strongly correlated. It is most suitable when the focus is on maximizing predictive accuracy rather than interpreting the relationships between individual predictors and the response variable.

#Question 3(b) - Ans

Principal Component Regression (PCR) has some key assumptions. It assumes that there is a linear relationship between the principal components and the outcome. It also assumes the residuals (errors) have constant variance, are independent, and are normally distributed. To check these, we can use scatter plots to verify linearity, and residual plots for constant variance. We can use residual plots to test for independence and Q-Q plots or histograms for checking normality.

PCR also needs the right number of components to be chosen, which can be done using scree plots or cross-validation.

#Question 3(c) - ans

For Principal Component Regression (PCR), the output can be present several way. First, explained variance would be shown using a scree plot, which helps illustrate how much each principal component captures the original data's variability. This allows stakeholders to understand which components are most important.

Additionally, model performance metrics such as PRESS, MSE or RMSE can be used to show how well/worse the model fits and how accurate it is. Residual analysis, including residual plots, can be used to verify any issues (i.e - difference from actual value, any particular group,overlap data etc). Finally, a clear interpretation summary would explain the key findings, predictors, and their impact on the outcome in plain English.

#Question 4(a) -

Supervised learning have both the input (predictors) and output (response) variables. The primary objective is to find a relationship between the inputs and outputs to make predictions for new data and/or understand the influence of predictors on the response variable. Regression and classification are supervised learning models. For example, predicting an spam email based on features like subject line and content is a supervised learning.

In terms of KNN classification, its a supervised machine learning algorithm. It works by looking at the closest k data points (neighbors) to the new point. Mostly its find distance based on Euclidean distance. The new point is assigned to the class that has the the most vote among other neighbors.If there is a tie, more than k neighbors might be considered. In some cases, closer neighbors are given more importance by using weight.To choose the best number of neighbors k, cross-validation methods are used. Also leave-one-out cross-validation is also used to calculate the best k. The best k is the one that give the highest accuracy or lowest error rate.

In Unsupervised learning, we do not have labeled data. This means we do not know the group each data point belongs to. The goal is to group the data based on how similar or close they are. Using a k-NN method for clustering, the algorithm groups points that are near each other, based on their distance, without needing labels.This is helpful in situations like market segmentation, where we want to group customers based on their behaviors (such as shopping habits), even though we don't know their spending category ahead of time.

In terms of k-Means clustering, its group the data points into k clusters based on how close they are to each other. The process starts by randomly assigning data points to k clusters. The average position of points in each cluster is called the centroid.Each data point is moved to the cluster with the nearest centroid, and the centroids are updated. This continues until no points need to be moved further. The goal of k-Means is to make the points in each cluster as close to their centroid as possible (Less variance within cluster). To decide the best number of clusters k, we can use the "elbow" method, where we plot the within-cluster sum of squares (WSS) against different numbers of clusters. The best k is where the curve forms an "elbow," showing a good balance.