Exercises: Prinicpal Component Analysis (PCA)

2023-09-26

Welcome to the exercise session on Principle Component Analysis. In this class we will go through the steps of conducting PCA on a dataset relating to kidney failure.

We will be using a couple of additional packages that are required to conduct PCA in R. The first is called 'corrr' and is used for running a PCA, and the second is 'corrplot', which plots the output of corrr.

First, read in the data and have a look at the columns that are available.

```r
kidney_data <- read.csv("../data/chronic_kidney_disease_full.csv")
```

Now we need to check whether there are NAs in our data - remember that this can break our analysis if we don't check for it first!

```r
colSums(is.na(kidney_data))
```

```
##       X     age      bp      sg      al      su     rbc      pc     pcc      ba
##       0       0       0       0       0       0       0       0       0       0
##     bgr      bu      sc     sod     pot    hemo     pcv    wbcc    rbcc     htn
##       0       0       0       0       0       0       0       0       0       0
##      dm     cad   appet      pe     ane   class no_name
##       0       0       0       0       0       0       0
```

```r
# is.na checks for NA values in each cell of the
# dataframe. colSums returns the sum of the columns - if any one entry is NA,
# then the colSums will give us an NA! So this is a neat way of checking for the
# presence of NA values.
```

Check which of the variables are categorical and which are continuous - remember that we need continuous data for PCA! Extract the columns that correspond to continuous data. How many variables do you get?

```r
head(kidney_data)
```

```
##   X age bp    sg al su    rbc       pc        pcc         ba bgr bu  sc sod pot
## 1 0  48 80 1.020  1  0      ?   normal notpresent notpresent 121 36 1.2   ?   ?
## 2 1   7 50 1.020  4  0      ?   normal notpresent notpresent   ? 18 0.8   ?   ?
## 3 2  62 80 1.010  2  3 normal   normal notpresent notpresent 423 53 1.8   ?   ?
## 4 3  48 70 1.005  4  0 normal abnormal    present notpresent 117 56 3.8 111 2.5
## 5 4  51 80 1.010  2  0 normal   normal notpresent notpresent 106 26 1.4   ?   ?
## 6 5  60 90 1.015  3  0      ?        ? notpresent notpresent  74 25 1.1 142 3.2
##   hemo pcv wbcc rbcc htn  dm cad appet  pe ane class no_name
## 1 15.4  44 7800  5.2 yes yes  no  good  no  no   ckd
## 2 11.3  38 6000    ?  no  no  no  good  no  no   ckd
## 3  9.6  31 7500    ?  no yes  no  poor  no yes   ckd
## 4 11.2  32 6700  3.9 yes  no  no  poor yes yes   ckd
## 5 11.6  35 7300  4.6  no  no  no  good  no  no   ckd
## 6 12.2  39 7800  4.4 yes yes  no  good yes  no   ckd
```

```r
# Select the conitnuous variables
data_for_PCA <- kidney_data %>% select(age:sg, bgr:rbcc)
```

```r
# Remove NA values, and apply character-to-numeric conversion
data_for_PCA <- data_for_PCA  %>%
  mutate_if(is.character, as.numeric) %>% na.omit()
```

```
## Warning in mask$eval_all_mutate(quo): NAs introduced by coercion

## Warning in mask$eval_all_mutate(quo): NAs introduced by coercion

## Warning in mask$eval_all_mutate(quo): NAs introduced by coercion

## Warning in mask$eval_all_mutate(quo): NAs introduced by coercion

## Warning in mask$eval_all_mutate(quo): NAs introduced by coercion

## Warning in mask$eval_all_mutate(quo): NAs introduced by coercion

## Warning in mask$eval_all_mutate(quo): NAs introduced by coercion

## Warning in mask$eval_all_mutate(quo): NAs introduced by coercion

## Warning in mask$eval_all_mutate(quo): NAs introduced by coercion

## Warning in mask$eval_all_mutate(quo): NAs introduced by coercion

## Warning in mask$eval_all_mutate(quo): NAs introduced by coercion

## Warning in mask$eval_all_mutate(quo): NAs introduced by coercion
```

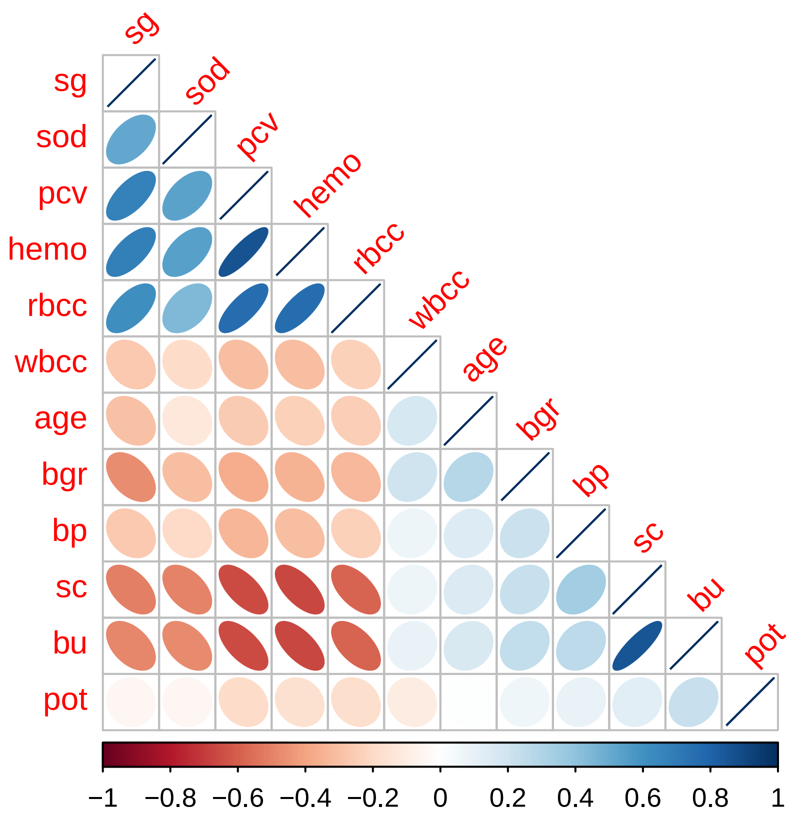
```r
# Scale (subtract mean and divide by SD) and convert to dataframe
data_for_PCA <- as.data.frame(scale(data_for_PCA))

head(data_for_PCA)
```

```
##            age         bp         sg        bgr          bu          sc
## 4  -0.24659614 -0.4270850 -2.4869076 -0.2786244  0.05994592  0.52602732
## 6   0.52059185  1.3068802 -0.7155313 -0.8483391 -0.61732456 -0.39100695
## 10  0.07306552  1.3068802  0.1701568 -0.9013358  1.17416510  1.68081121
## 12  0.71238885 -0.4270850 -1.6012195  3.2059100  0.14733566  0.15242077
## 13  1.03205051 -0.4270850 -0.7155313  0.9270510  0.40950488 -0.05136463
## 15  1.03205051  0.4398976 -1.6012195  0.2513428  0.80275870  0.62792001
##           sod        pot       hemo        pcv       wbcc       rbcc
## 4  -4.0111323 -0.6705286 -0.7568731 -0.9702043 -0.6465486 -0.8874759
## 6   0.4763442 -0.4436665 -0.4039091 -0.1868746 -0.2707696 -0.3881477
## 10 -3.5768604 -0.2816220 -1.3569120 -1.3059170  1.1981847 -1.0872071
## 12 -1.1159861 -0.1195776 -0.8980588 -0.9702043 -1.3981066 -0.9873415
## 13 -0.1026850  0.3989645 -1.2863192 -1.4178213  1.2323465 -1.3868040
## 15 -1.2607434  0.5934178 -2.7334718 -2.7606722  0.8224057 -2.1857291
```

Now plot a correlation matrix with your standardised data. For a very simple correlation matrix you can use the cor function from the corrr package and then use corrplot, but you could also try creating a correlation matrix in ggplot.

```r
# Correlation matrix
corr_matrix <- cor(data_for_PCA)
```

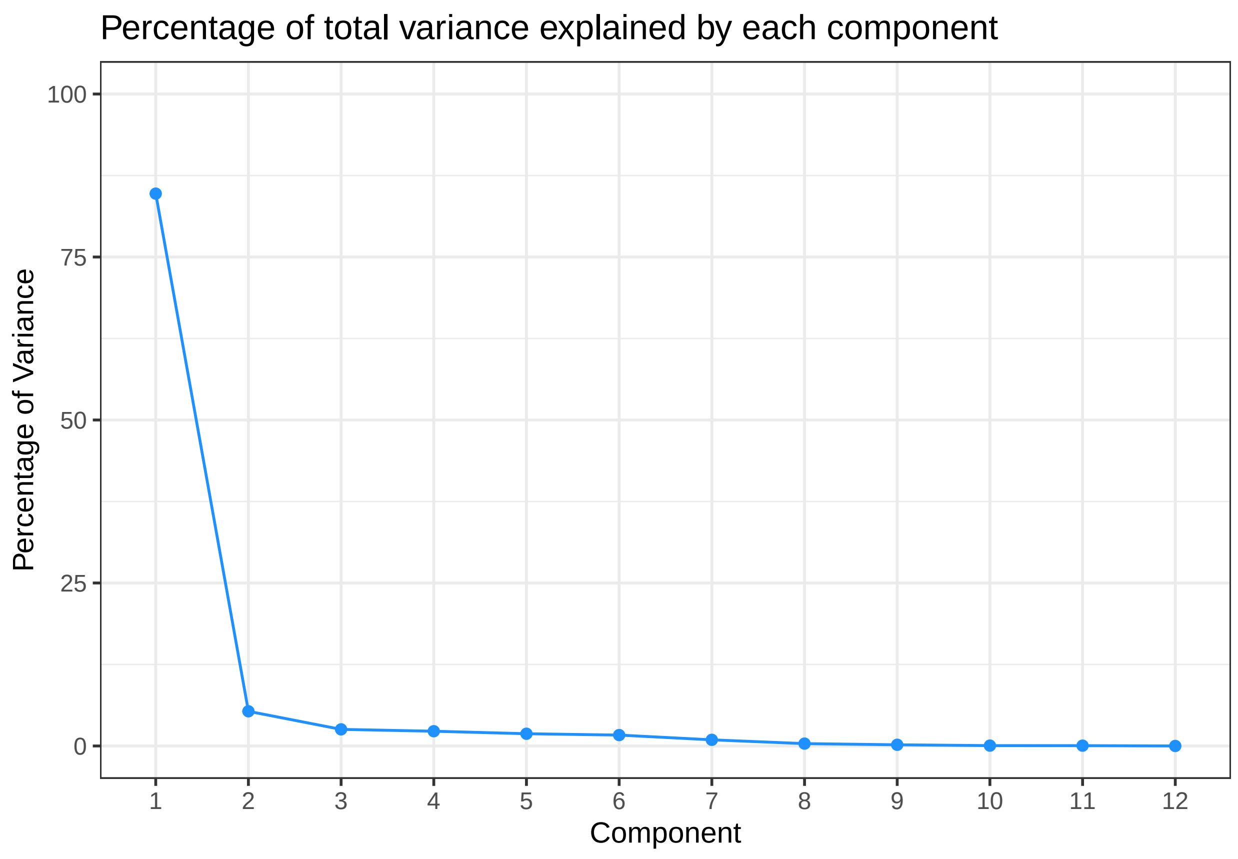
```r
# Correlation matrix plot
corrplot(
  corr_matrix,
  # Only plot the lower triangular
  type = 'lower',
  # Visualisation method: show ellipses
  # eccentricity scaled to the correlation value
  method = 'ellipse',
  # Sort by angular order of the eigenvectors
  order = 'AOE',
  # Rotate the text labels
  tl.srt = 45
)
```



Now, we can perform the PCA. We can use the command prcomp to perform the PCA on the scaled data.

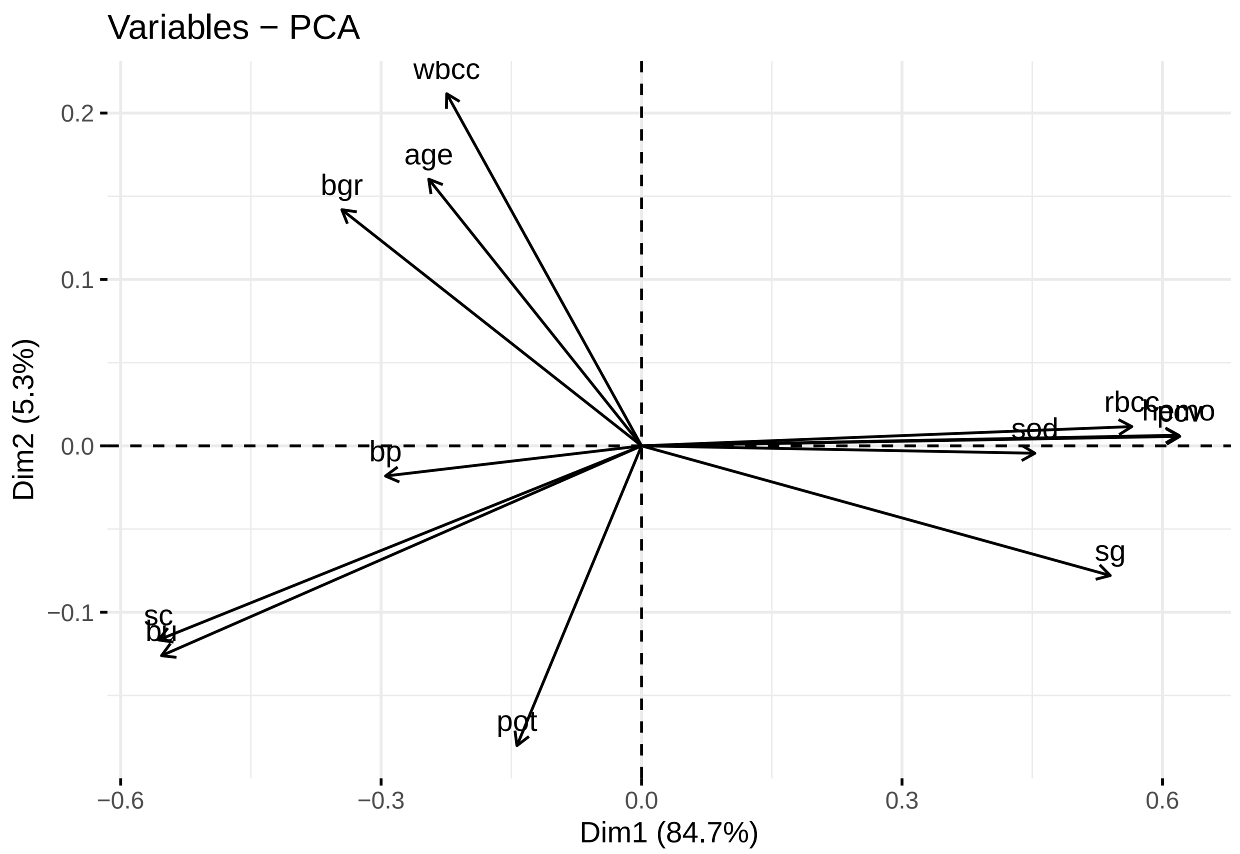
```r
# Perform PCA
PCA <- prcomp(corr_matrix)

# SD of the principal components
sdevs <- PCA$sdev

# Total variance
total_var <- sum(sdevs^2)

# Get proportion variance (divide SD^2 by total variance)
proportion_of_variance <- data.frame(Proportion_of_Variance = (sdevs^2) / total_var * 100)
```

```r
proportion_of_variance$Component <- as.character(row.names(proportion_of_variance))


# Scree plot
scree_plot <- ggplot(
  proportion_of_variance, aes(
    fct_reorder(Component, -Proportion_of_Variance), Proportion_of_Variance, group = 1)
  ) +
  # Add line plot
  geom_line(color = 'dodgerblue') +
  # Add scatter plot
  geom_point(color = 'dodgerblue') +
  # Change y limit, xy labels, title, and theme
  ylim(0, 100) +
  xlab('Component') +
  ylab('Percentage of Variance') +
  ggtitle('Percentage of total variance explained by each component') +
  theme_bw()

scree_plot
```



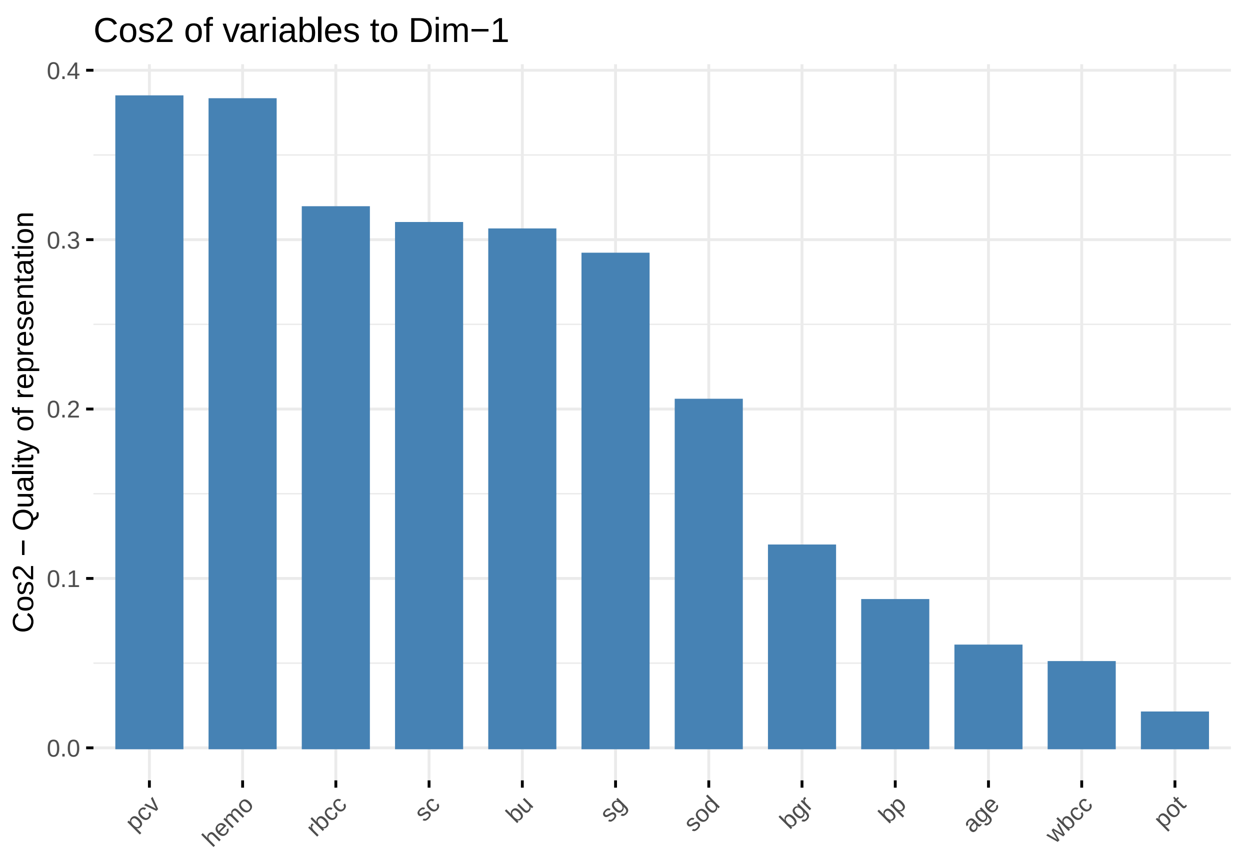
We can also use fviz_pca_var to plot the each variable in terms of the principal components.

```
fviz_pca_var(PCA, col.var = 'black')
```



Finally, we can use a cos2 plot to transform the principal components back into our original:

```
fviz_cos2(PCA, choice = "var", axes = 1)
```



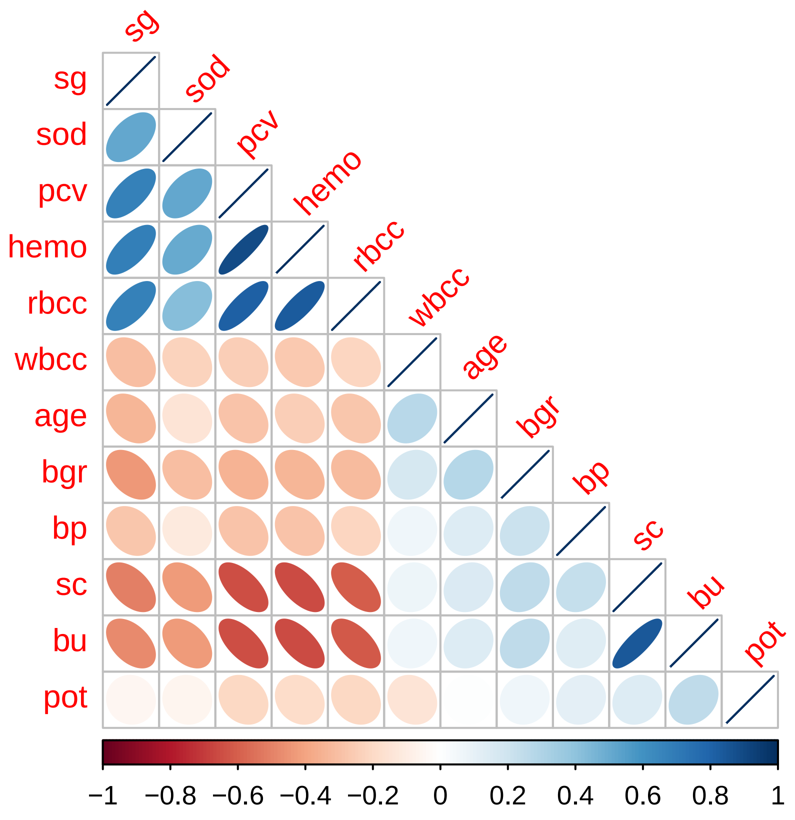
We can also use PCA to predict whether or not a patient will have chronic kidney disease. We split the data into a train set, on which we perform the PCA, and a test set, for which we try to predict whether each patient has chronic kidney disease.

We can then plot the principal components for each of these patients on a scatter plot, and see whether the patients are separated by their kidney condition.

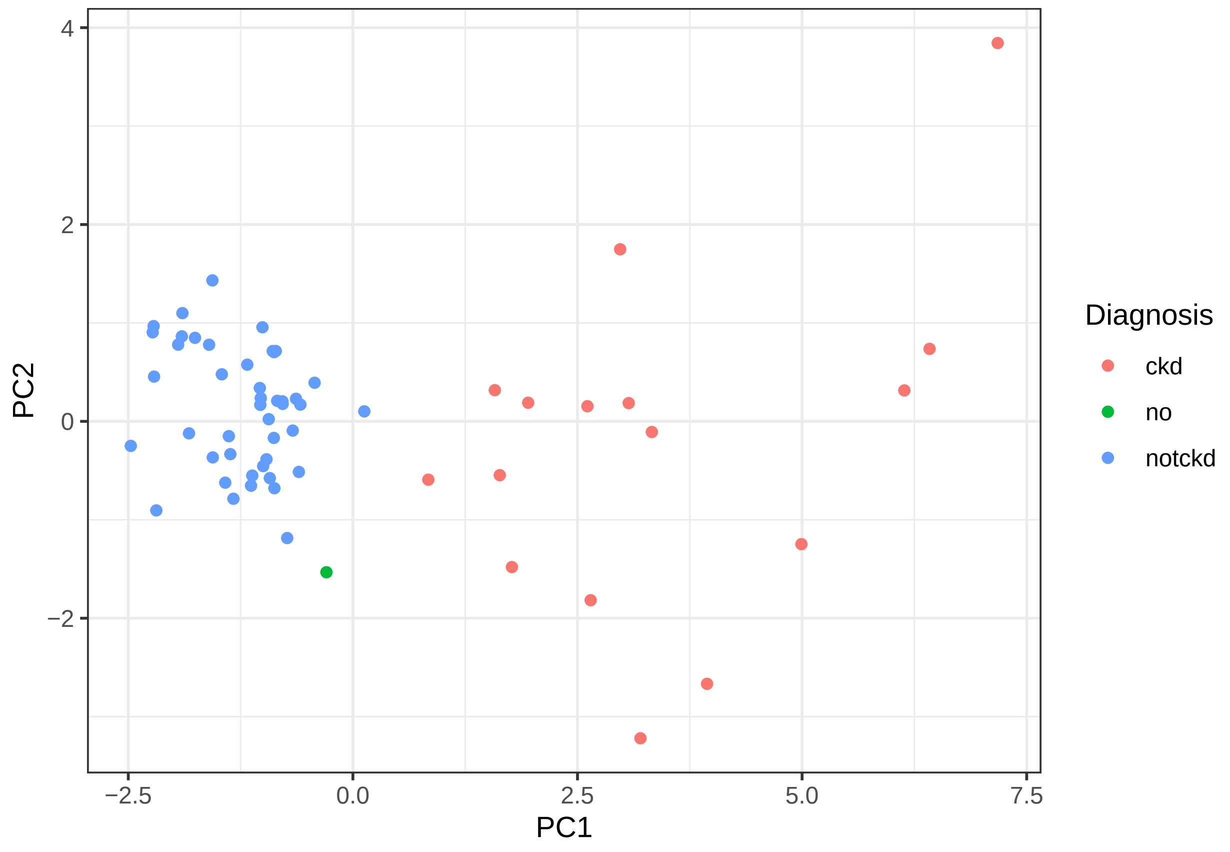
```r
# Take the continuous data
data_for_prediction <- kidney_data %>% select(X, age:sg, bgr:rbcc)

# Remove NAs
data_for_PCA <- data_for_prediction %>%
  mutate_if(is.character, as.numeric) %>% na.omit()
```

```
## Warning in mask$eval_all_mutate(quo): NAs introduced by coercion

## Warning in mask$eval_all_mutate(quo): NAs introduced by coercion

## Warning in mask$eval_all_mutate(quo): NAs introduced by coercion

## Warning in mask$eval_all_mutate(quo): NAs introduced by coercion

## Warning in mask$eval_all_mutate(quo): NAs introduced by coercion

## Warning in mask$eval_all_mutate(quo): NAs introduced by coercion

## Warning in mask$eval_all_mutate(quo): NAs introduced by coercion
## Warning in mask$eval_all_mutate(quo): NAs introduced by coercion

## Warning in mask$eval_all_mutate(quo): NAs introduced by coercion

## Warning in mask$eval_all_mutate(quo): NAs introduced by coercion

## Warning in mask$eval_all_mutate(quo): NAs introduced by coercion

## Warning in mask$eval_all_mutate(quo): NAs introduced by coercion
```

```r
# Add the category labels, i.e. does each person have chronic kidney disease
data_for_PCA$class <- kidney_data %>% filter(X %in% data_for_PCA$X) %>%
  select(class)

# Split the data into testing and training - choose a random sample from the full
# dataset
train <- data_for_PCA %>% sample_frac(0.70)
test  <- dplyr::anti_join(data_for_PCA, train, by = 'X')

#Scale the training set and produce a correlation matrix
scale_train <- as.data.frame(scale(train %>% select(age:sg, bgr:rbcc)))
corr_matrix <- cor(scale_train)
corrplot(corr_matrix, type = 'lower', method = 'ellipse', order = 'AOE',
         tl.srt = 45)
```



```r
# Perform the PCA on the scaled training set
PCA <- prcomp(scale_train, scale. = TRUE, center = TRUE)

# Scale and perform PCA on the test set
scale_test <- as.data.frame(scale(test %>% select(age:sg, bgr:rbcc)))
pred <- predict(PCA, newdata = scale_test)
train_df <- as.data.frame(PCA$x[, 1:2])
train_df$Diagnosis <- unname(unlist(train$class))

# Transform the testing data using the principal components.
predict_df <- as.data.frame(pred[, 1:2])
predict_df$Diagnosis <- unname(unlist(test$class))

# Plot the results
scatter <- ggplot(predict_df, aes(x = PC1, y = PC2, color = Diagnosis)) +
  geom_point() + theme_bw()
scatter
```


PC2
PC1
Diagnosis
ckd
no
notckd