R Exercise 4: INTA 6450 - Data Analytics and Security

R Exercise 4

Prompt

1. Complete the following exercise:

- `clusters.r`

2. You may make changes to the code to explore how they work
3. Then, submit the output for running these commands. You may copy/paste output from your terminal into a word or text document. You can convert the notebook to PDF submission:File->Compile Report->Report output format as HTML/PDF.
4. Write 100-150 words describing one extension you made to the code. You may post the change description as a comment at the beginning of your code, or submit the write-up in a separate document. Make sure to clearly mark where in your code the changes were made either by commenting, highlighting, or noting the command lines where the changes occur.

Submissions

You will submit the following:

1. Original code output (R file code and output)
2. Your code change and output from that code change (comment where your code changes start and end)
3. Your code change description (this should be between 100 to 150 words)

An example of what the change description should look like:

```
##############################
# CHANGE DESCRIPTION
# 100-150 words on the changes I've made, including a reference to which
# initial command I've changed >
# END OF CHANGE DESCRIPTION
##############################
```

Files must be in one of the following formats: pdf, doc, html

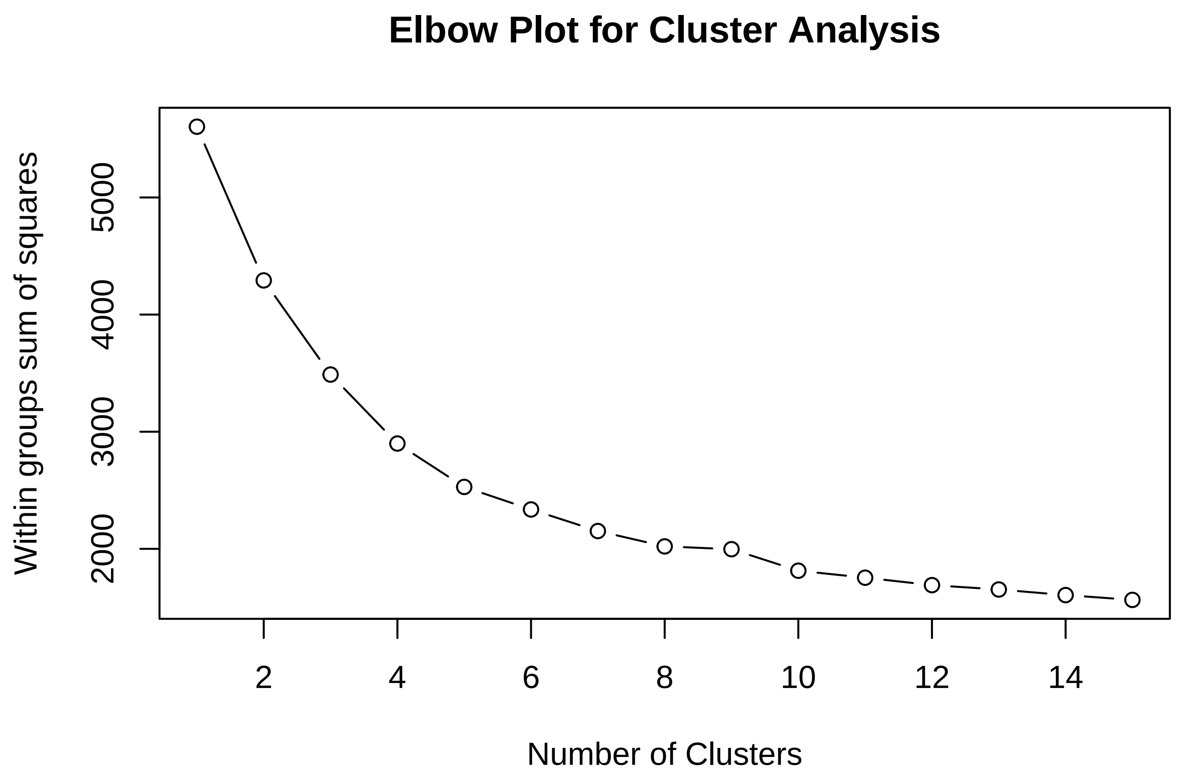
Please note some of these exercises have graphs/charts - be sure when you export the original output to also check that the file includes graphs/charts that may have been present. If it does not, you will need to re-export or manually include the graphs in your final pdf, doc or html file

Solution Summary

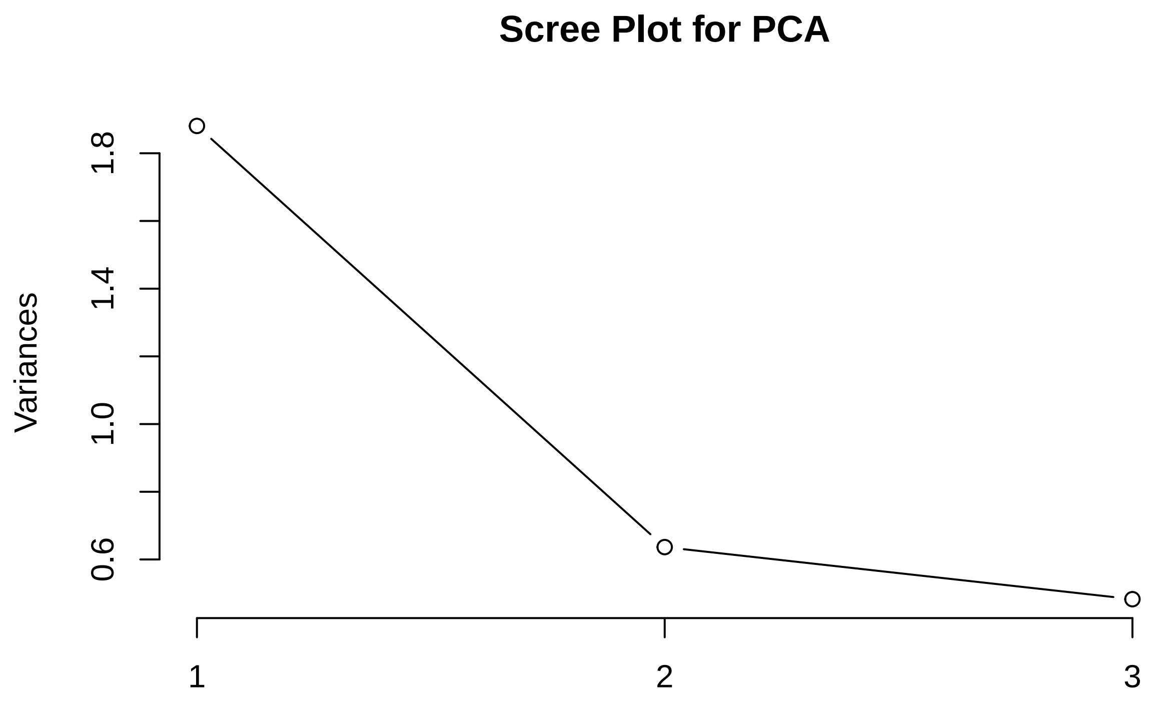
To visually analyze some of the modeling performed, several plots were developed. An elbow plot is create for cluster analysis to determine
the optimal number of clusters. Elbow plots provide a visual representation of the within-cluster sum of squares, allowing to identify where the addition of more clusters results in less return. A scree plot is created for PCA to determine the number of principal components. Scree plots are used for helping us decide how many components to retain for further analysis. Last, bar plots are generated to visually compare R-squared values from different models. R-squared values are used to evaluate the goodness of fit of a model. Comparison allows us to evaluate incorporating multiple variables versus a single principal component.

The code changes are attached below and show the code for each plot. The original code output is also included for reference.

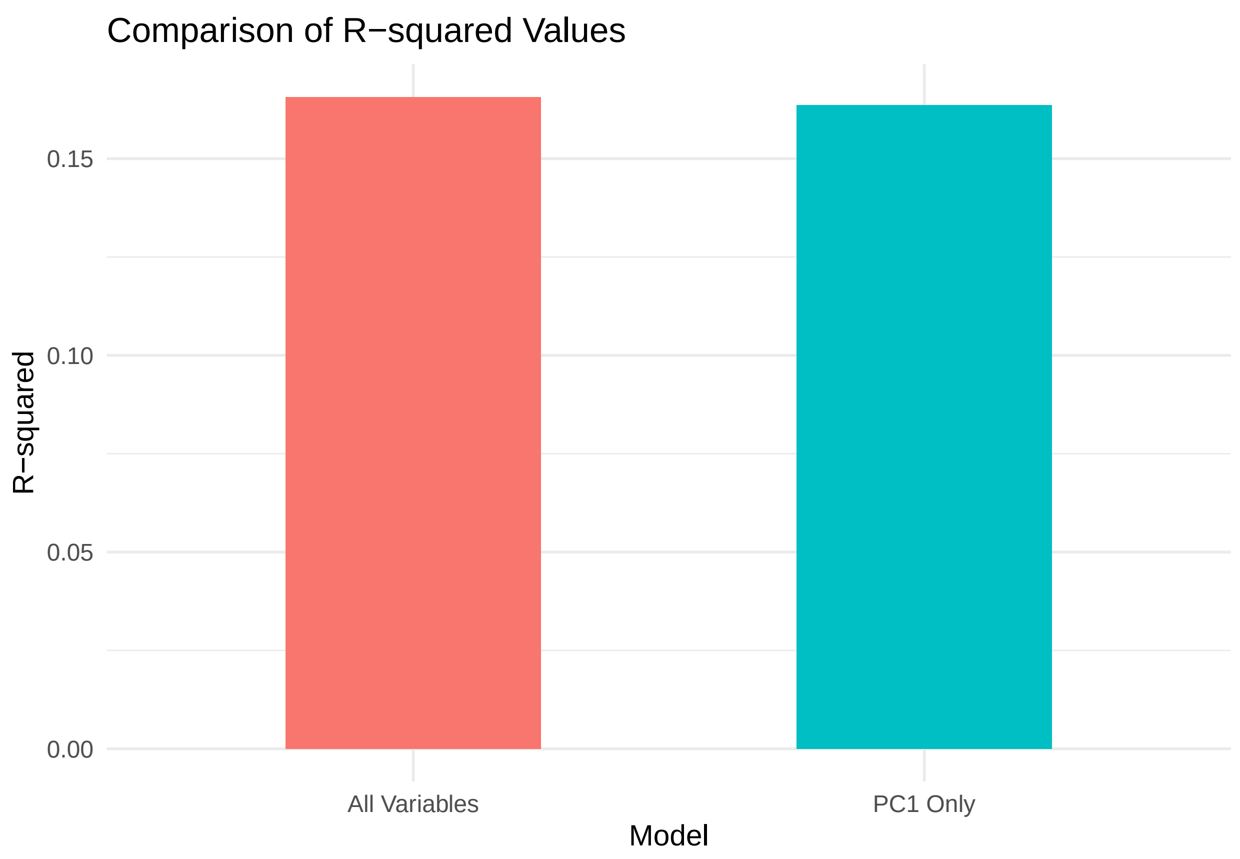
```r
# Elbow plot for cluster analysis
wss <- (nrow(subwages2) - 1) * sum(apply(subwages2, 2, var))
for (i in 2:15) {
  wss[i] <- sum(kmeans(subwages2, centers=i, nstart=5)$tot.withinss)
}
plot(1:15, wss, type="b", xlab="Number of Clusters", ylab="Within groups sum of squares", main="Elbow Pl
```



```r
# Scree plot for PCA
plot(principal.components, type="lines", main="Scree Plot for PCA")
```



```r
# Extract R-squared values
r_squared_values <- data.frame(
  Model = c("PC1 Only", "All Variables"),
  R_squared = c(summary(one.pc)$r.squared, summary(all.variables)$r.squared)
)

# Bar plot for R-squared values comparison
r_squared_plot <- ggplot(r_squared_values, aes(x = Model, y = R_squared, fill = Model)) +
  geom_bar(stat = "identity", width = 0.5) +
  labs(title = "Comparison of R-squared Values",
       x = "Model",
       y = "R-squared") +
  theme_minimal() +
  theme(legend.position = "none")

# Display the plot
print(r_squared_plot)
```



```r
# Save the plot
ggsave("r_squared_comparison_plot.png", plot = r_squared_plot, width = 6, height = 4, dpi = 300)
```

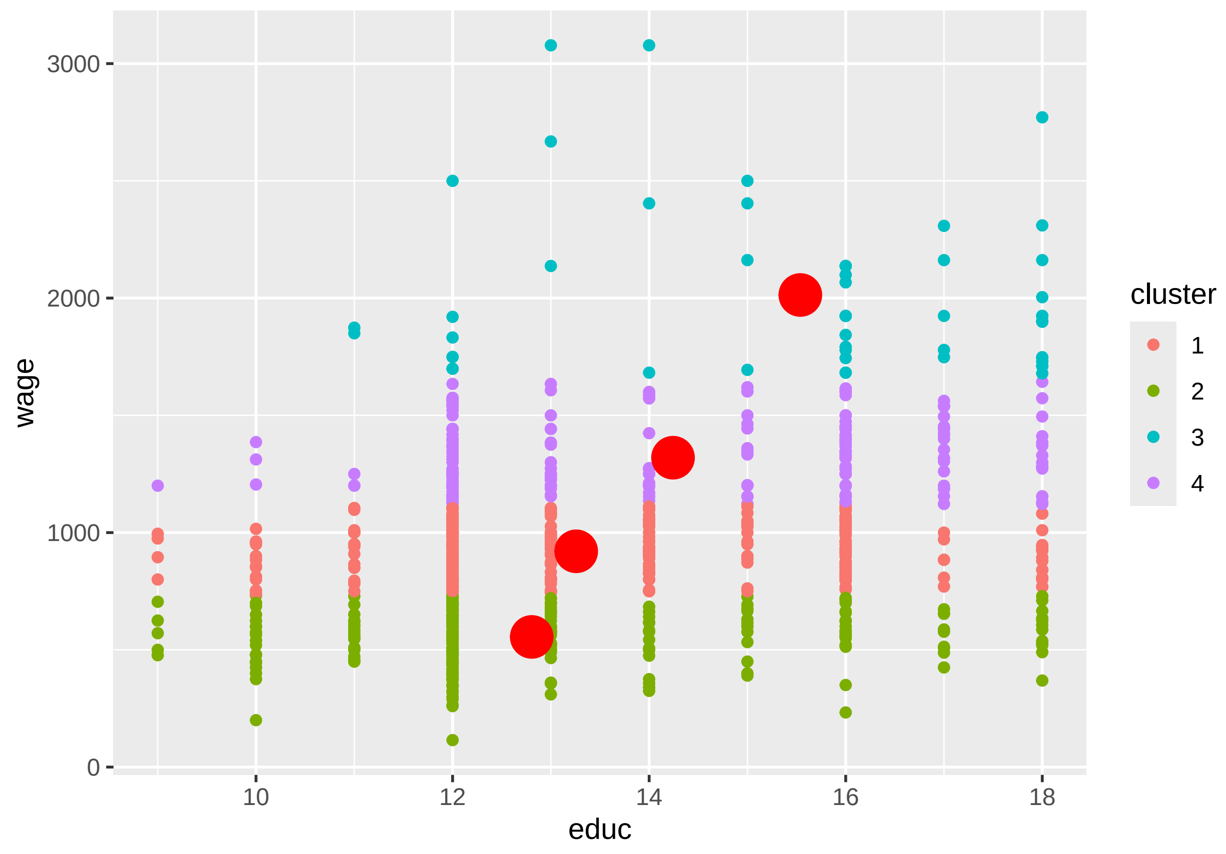
Original Code

```r
# Clustering example
library(MASS)
library(ggplot2)
wages<-read.csv('http://inta.gatech.s3.amazonaws.com/wage2.csv')
subwages <-wages[,c("wage","IQ","KWW","educ","age","married")]
# Cluster based only on a subset of the data
cluster.results<-kmeans(subwages, 4, nstart=5)
# Calculate clusters, using 5 different starting samples
print(cluster.results)
```

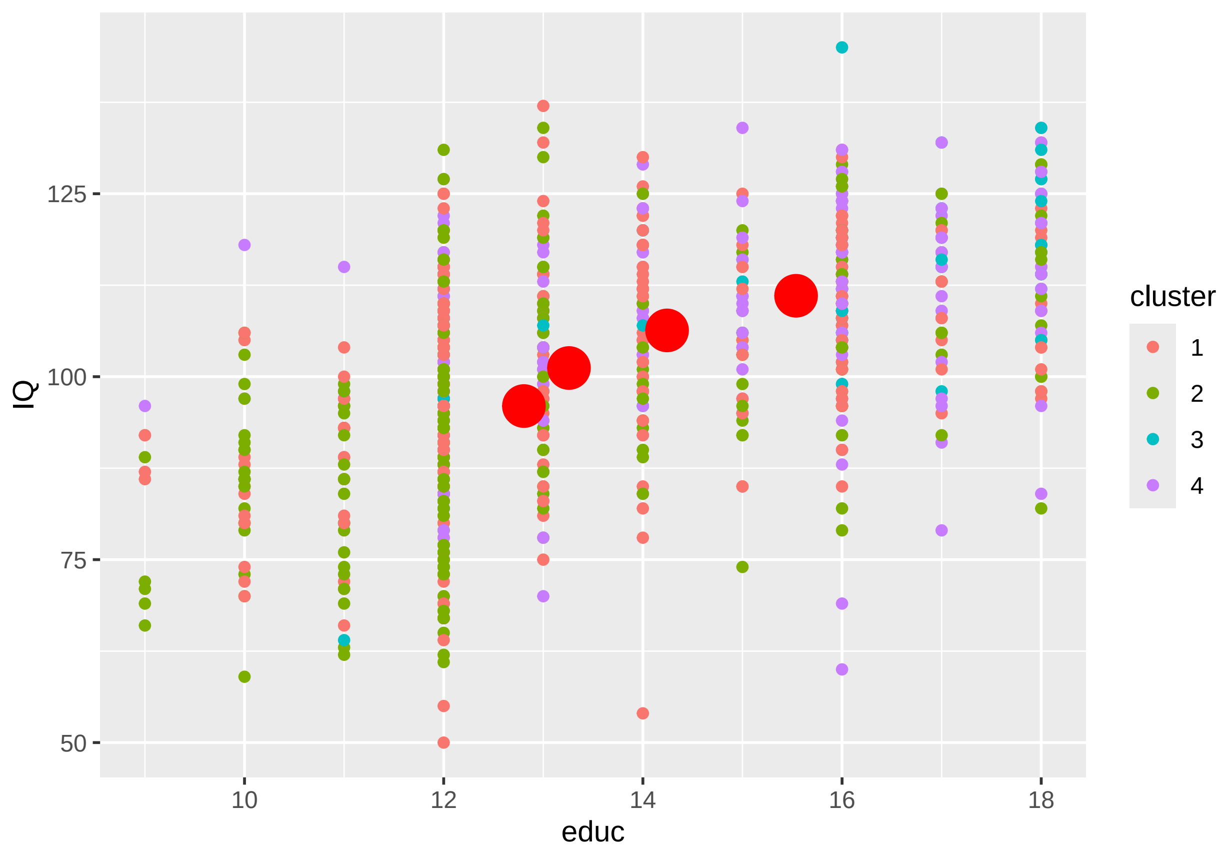
```
## K-means clustering with 4 clusters of sizes 376, 293, 52, 214
##
## Cluster means:
##        wage        IQ      KWW     educ      age   married
## 1  920.2074 101.18617 35.73936 13.25798 33.30585 0.9069149
## 2  555.1092  95.98976 32.86007 12.80546 32.38567 0.8293515
## 3 2013.2692 111.05769 41.38462 15.53846 34.28846 0.9230769
## 4 1319.3645 106.32243 38.33178 14.24299 33.34112 0.9485981
##
## Clustering vector:
##   [1] 1 1 1 2 2 4 2 1 4 1 1 1 1 4 3 1 4 1 1 2 4 4 1 3 1 4 4 1 4 2 1 4 3 2 3 1 4
##  [38] 4 2 1 2 4 4 1 3 1 1 4 4 1 1 4 3 1 2 1 1 4 4 1 3 4 1 1 1 2 2 2 4 4 4 3 3 4
##  [75] 1 1 1 1 2 1 1 2 4 3 2 1 1 1 2 2 3 1 1 4 1 2 4 4 1 1 4 1 1 2 4 1 4 1 4 1 1
## [112] 1 4 1 2 2 2 2 4 2 4 4 1 1 1 1 1 3 4 1 4 4 4 2 4 1 1 1 1 1 1 2 1 2 1 4 3 1
## [149] 1 4 4 2 2 4 2 1 3 3 1 4 4 1 1 2 4 4 3 4 4 2 1 1 1 1 1 2 4 2 2 2 2 1 4 1 4
## [186] 2 2 1 2 1 1 1 1 3 4 4 2 3 3 1 1 2 2 2 2 1 2 2 2 1 1 4 2 2 2 4 1 2 4 2 1 2
## [223] 1 4 1 4 1 4 2 2 4 4 4 4 1 1 2 2 3 3 4 3 1 1 2 4 1 1 1 1 4 4 1 4 1 1 1 2 2
## [260] 1 1 4 2 2 1 1 2 3 1 2 4 4 1 4 2 4 1 1 2 1 1 4 4 4 3 1 1 1 2 2 2 4 1 1 1 4
## [297] 2 1 1 1 1 2 1 1 3 4 2 1 1 1 1 2 1 1 4 1 4 1 1 4 1 2 1 3 1 2 1 2 4 1 1 2 4
## [334] 4 4 2 1 1 1 2 4 1 4 1 2 1 1 2 1 2 4 4 1 4 4 1 2 1 1 2 1 1 2 2 1 2 2 1 2 1
## [371] 4 2 1 1 1 4 1 2 1 4 1 1 4 2 2 2 1 1 1 1 1 1 1 2 1 2 1 4 1 1 4 2 2 2 2 2 2
## [408] 2 1 4 4 4 4 4 1 1 4 2 4 1 3 4 1 4 2 4 1 2 4 2 3 2 1 2 1 4 3 2 4 4 2 1 1 1
## [445] 2 4 4 4 2 2 3 1 2 1 1 1 2 1 3 2 1 4 2 1 1 2 2 1 1 4 2 2 4 4 4 1 4 4 1 2 4
## [482] 1 2 2 2 1 1 2 1 1 1 2 4 1 4 2 1 2 1 1 1 2 1 4 1 2 1 1 1 2 1 2 1 2 2 2 1 2
## [519] 4 4 4 2 1 4 1 2 1 2 4 1 1 2 2 2 2 4 2 2 1 2 4 4 2 2 2 3 1 4 1 4 4 4 1 2 1
## [556] 1 1 1 4 3 2 2 2 1 4 2 2 1 1 2 2 2 2 4 1 1 2 4 1 2 4 4 4 2 2 2 2 2 2 2 4 4
## [593] 4 2 4 1 1 1 1 4 2 4 3 1 1 2 1 2 1 3 1 1 2 2 1 3 1 4 2 2 4 2 2 1 4 4 4 4 2
## [630] 1 3 1 2 1 3 2 2 2 1 2 2 4 2 4 4 4 1 2 2 1 4 1 2 2 1 2 4 1 1 1 1 4 1 2 1 2
## [667] 4 1 2 1 1 2 2 2 2 2 2 2 2 1 2 2 1 1 1 4 4 2 1 2 1 4 1 1 1 1 2 4 1 2 3 2 3
## [704] 1 3 4 2 4 4 1 4 1 1 1 4 4 1 1 1 1 2 4 2 4 1 4 2 4 3 1 3 2 1 3 2 1 1 1 4 2
## [741] 4 2 1 1 4 4 3 4 1 2 3 3 1 1 2 2 1 4 4 1 3 1 3 4 1 1 1 3 4 1 1 1 4 1 4 1 4
## [778] 1 1 1 4 4 1 2 2 1 1 1 2 3 1 1 1 1 1 2 1 1 1 1 4 2 1 1 1 1 4 2 1 1 1 4 1 2
## [815] 2 2 1 3 4 2 2 4 1 1 2 4 2 1 1 4 1 1 2 1 2 1 4 2 2 2 2 2 2 4 2 1 2 2 1 2 2
## [852] 2 2 2 2 2 1 1 2 1 1 1 1 1 1 4 2 2 2 2 2 1 4 2 1 1 2 2 1 2 1 1 2 2 2 1 2 1
## [889] 4 2 2 1 2 1 1 2 2 1 1 1 4 2 2 2 1 2 1 2 4 2 2 4 1 1 1 2 2 2 2 4 2 1 2 2 4
## [926] 2 1 2 2 2 2 4 2 1 1
##
## Within cluster sum of squares by cluster:
## [1] 4231857 4165182 6153621 4471404
##  (between_SS / total_SS =  87.6 %)
##
```

```
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
## [6] "betweenss"    "size"         "iter"         "ifault"
```

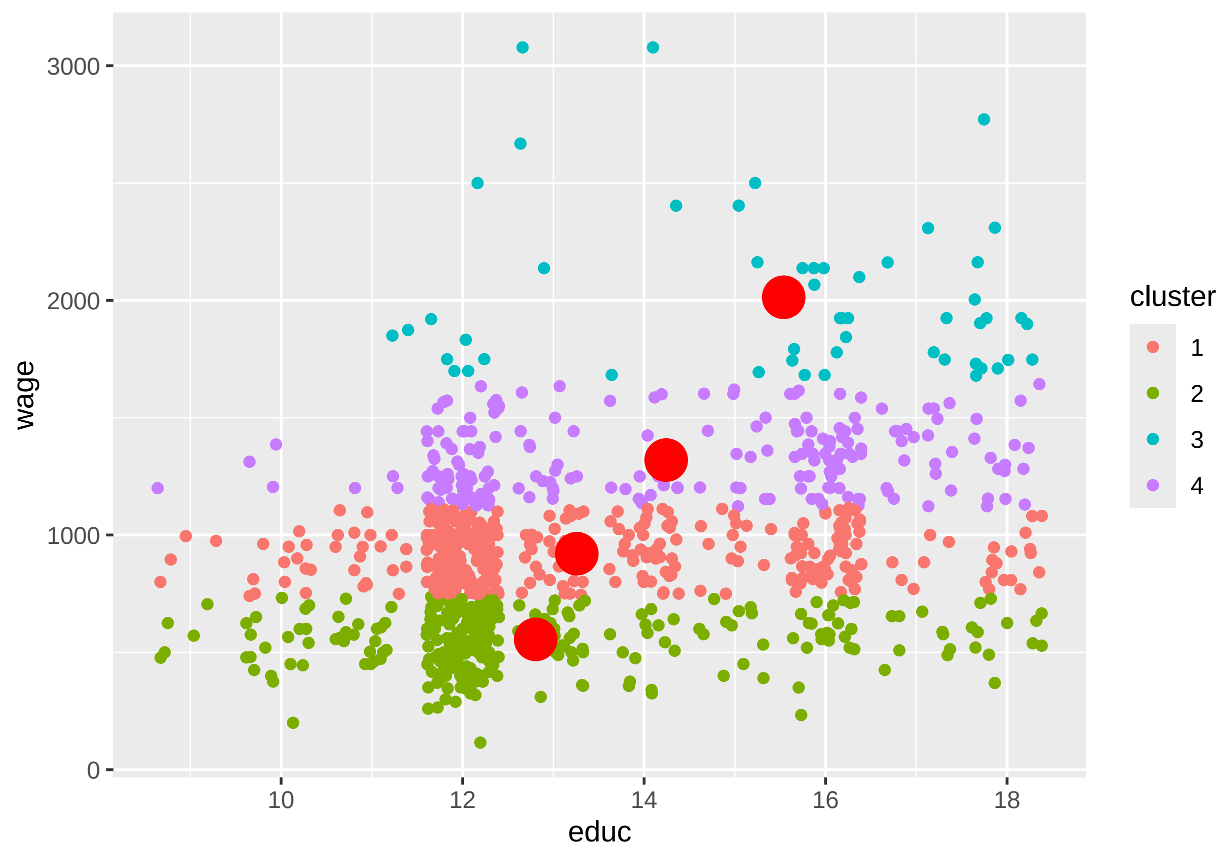
```
# Look at cluster results
subwages$cluster<-as.factor(cluster.results$cluster)
# The 'cluster' element of the results is the cluster which each data
# point belongs

ggplot(data=subwages, aes(x=educ, y=wage)) + geom_point(aes(colour=cluster))  +
  geom_point(data=data.frame(cluster.results$centers), colour='red', size=7)
```



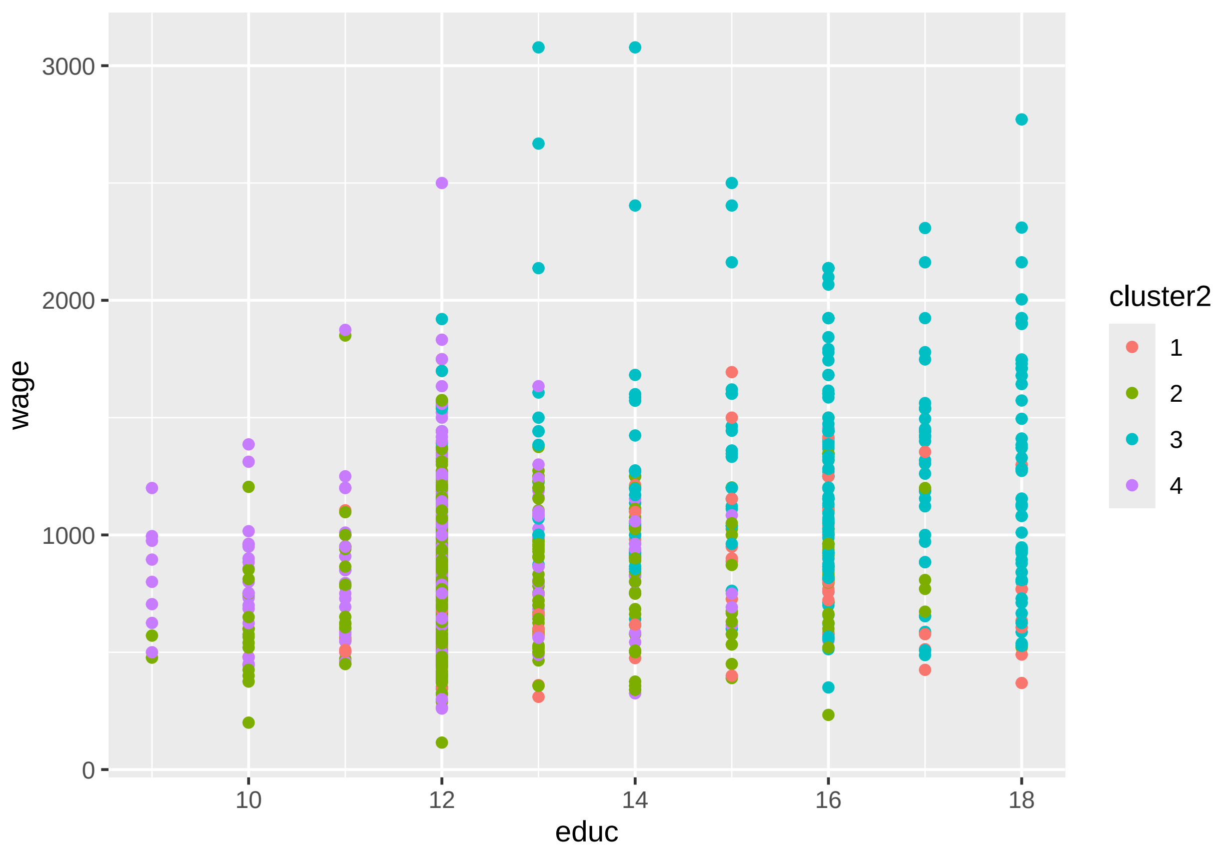
```
ggplot(data=subwages, aes(x=educ, y=IQ)) + geom_point(aes(colour=cluster)) +
  geom_point(data=data.frame(cluster.results$centers), colour='red', size=7)
```



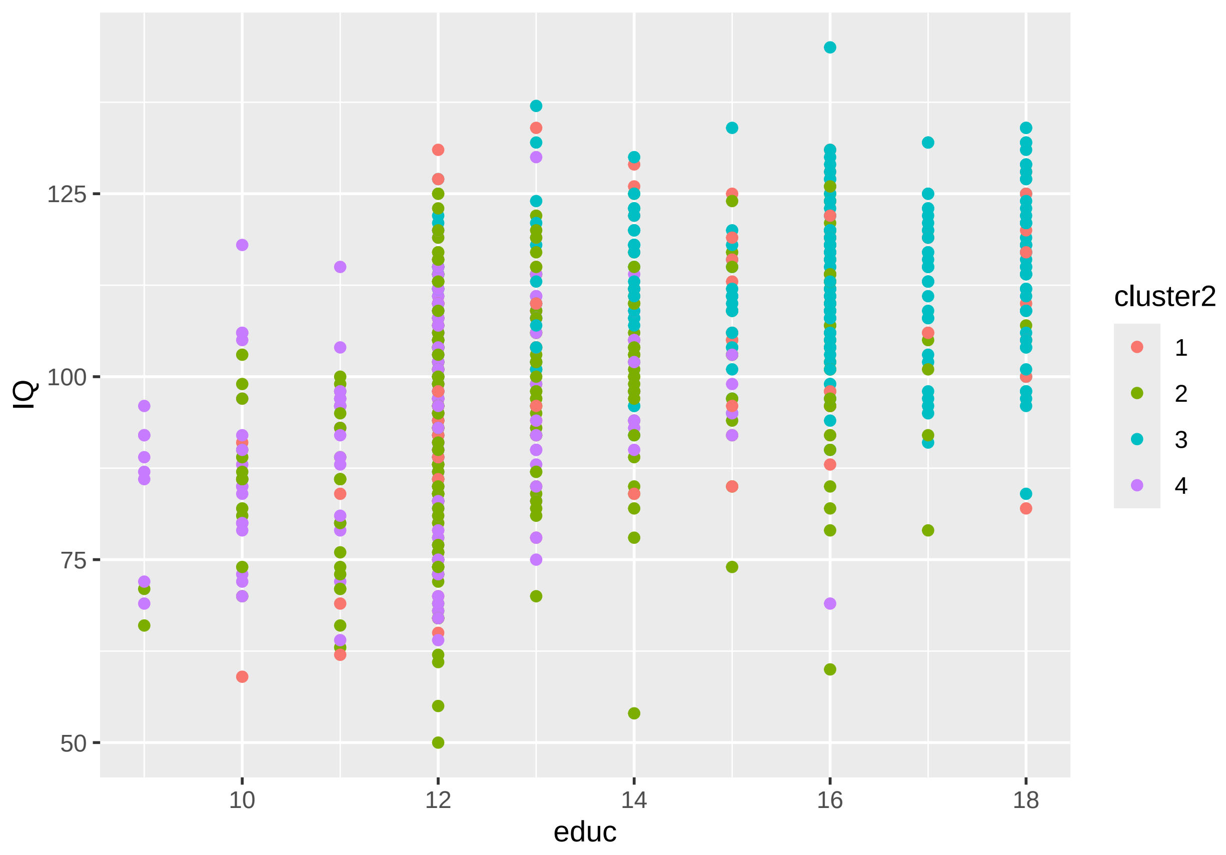
```r
# We can plot the data points with colors for the different clusters.
# The colors match up differently along different dimensions

# If you want to see these plots with the data points less on top of each other,
# 'geom_jitter' adds soem noise to each point to move them off of each other:
ggplot(data=subwages, aes(x=educ, y=wage)) + geom_jitter(aes(colour=cluster)) +
  geom_point(data=data.frame(cluster.results$centers), colour='red', size=7)
```



```
# When you look at these graphs, it's clear that clustering is mostly happening
# on wage. Why is that? How could you fix it?

####################
# Standardize before doing clustering
zscore<-function(x){
  out<-(x - mean(x))/sd(x)
  return(out)
}
subwages2<-subwages
subwages2$cluster<-NULL
for (col in names(subwages2)){
  subwages2[,col]<-zscore(subwages2[,col])
}
# This loop standardizes the data (subtracts mean and divides by standard deviation)
# so that the data is in z-scores

cluster.results2<-kmeans(subwages2, 4, nstart=5)
subwages$cluster2<-as.factor(cluster.results2$cluster)
ggplot(data=subwages, aes(x=educ, y=wage)) + geom_point(aes(colour=cluster2))
```



```r
ggplot(data=subwages, aes(x=educ, y=IQ)) + geom_point(aes(colour=cluster2))
```



```r
# Plot the original data using the new clusters - note there is more of an IQ-based
# pattern and less a pure wages one.

# Explore clusters in this data:
# 1) What happens if you use a different number of clusters?
# 2) How many clusters do you think there should be?
# 3) How good a fit can you get of wages using these clusters?

##########
# Principal Components Analysis
wages<-read.csv('http://inta.gatech.s3.amazonaws.com/wage2.csv')
subwages <-wages[,c("IQ","KWW","educ")]
# Create a subset of wages with just a few columns
principal.components <- prcomp(subwages, retx=T, center=T, scale=T)
# Do a principal components analysis on just the columns in subwages
print(principal.components$rotation) # The weights on each variable
```

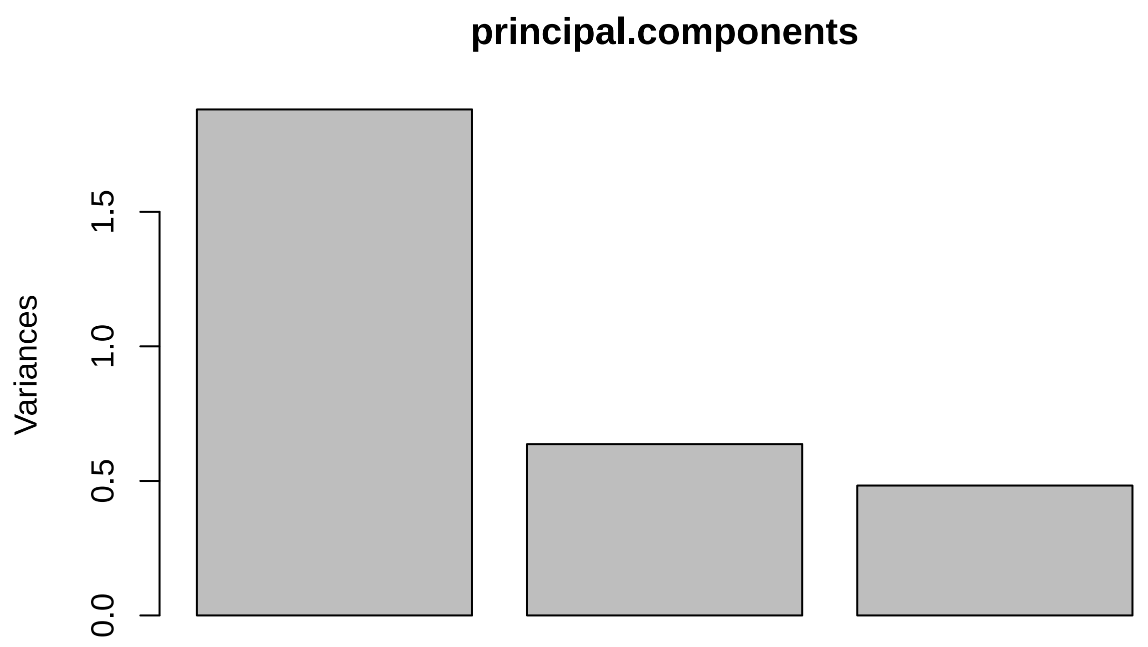
```
##            PC1        PC2         PC3
## IQ   0.5994431 -0.3116398 -0.73725745
## KWW  0.5413277  0.8363384  0.08661666
## educ 0.5896035 -0.4510196  0.67003657
```

```r
summary(principal.components) # The proportion of variance explained
```

```
## Importance of components:
```

```
##                           PC1    PC2    PC3
## Standard deviation     1.3714 0.7979 0.6948
## Proportion of Variance 0.6269 0.2122 0.1609
## Cumulative Proportion  0.6269 0.8391 1.0000
```

```
plot(principal.components) # Graphical depiction of proportion of variance
```



```
sw <- cbind(wages,data.frame(principal.components$x))
# Add the new principal components as columns to a new data frame, sw
# (New so we don't muck up going back and running other stuff on wages now)
one.pc <- lm(wage ~ PC1, data= sw)
all.variables <- lm(wage ~ IQ + KWW + educ, data= sw)
summary(one.pc)
```

```
##
## Call:
## lm(formula = wage ~ PC1, data = sw)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -914.52 -241.02  -35.99  194.77 2215.61
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  957.945     12.100   79.17   <2e-16 ***
## PC1          119.264      8.828   13.51   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 370 on 933 degrees of freedom
## Multiple R-squared:  0.1636, Adjusted R-squared:  0.1627
## F-statistic: 182.5 on 1 and 933 DF,  p-value: < 2.2e-16
```

```r
summary(all.variables)
```

```
##
## Call:
## lm(formula = wage ~ IQ + KWW + educ, data = sw)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -927.30 -240.12  -32.46  199.57 2250.92
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -231.6018    92.1449  -2.513 0.012123 *
## IQ             3.5675     0.9749   3.660 0.000267 ***
## KWW           10.6471     1.7859   5.962 3.54e-09 ***
## educ          33.2366     6.5997   5.036 5.71e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 369.9 on 931 degrees of freedom
## Multiple R-squared:  0.1657, Adjusted R-squared:  0.163
## F-statistic: 61.64 on 3 and 931 DF,  p-value: < 2.2e-16
```

```r
# How much better does having three variables do compared to the one principal
# component here? (Look at R^2)

# Try to replicate this using more variables, including perhaps age and marital
# Status. Would you see the same thing?
```