Unsupervised Learning

2023-11-28

```
condensed_data = read.csv("condensed_train_data.csv")
```

```
counts = c(nrow(condensed_data[condensed_data[, 18] == 0.5, ]), nrow(condensed_data[condensed_data[, 18]

scores = c(0.5, 1, 1.5, 2, 2.5, 3, 3.5, 4, 4.5, 5, 5.5, 6)

countsdf = data.frame(scores=scores, counts=counts)
countsdf
```

```
##    scores counts
## 1     0.5      4
## 2     1.0     29
## 3     1.5     53
## 4     2.0     71
## 5     2.5    155
## 6     3.0    271
## 7     3.5    384
## 8     4.0    410
## 9     4.5    320
## 10    5.0    149
## 11    5.5     99
## 12    6.0     31
```

```
set.seed(129)
kmeans_res = kmeans(condensed_data[, -c(1, 2)], centers=12)

cluster_labels <- kmeans_res$cluster
centroids <- kmeans_res$centers

plot_data <- data.frame(condensed_data, Cluster = as.factor(kmeans_res$cluster))
centroids_df <- data.frame(centroids, Cluster = factor(1:12))
```

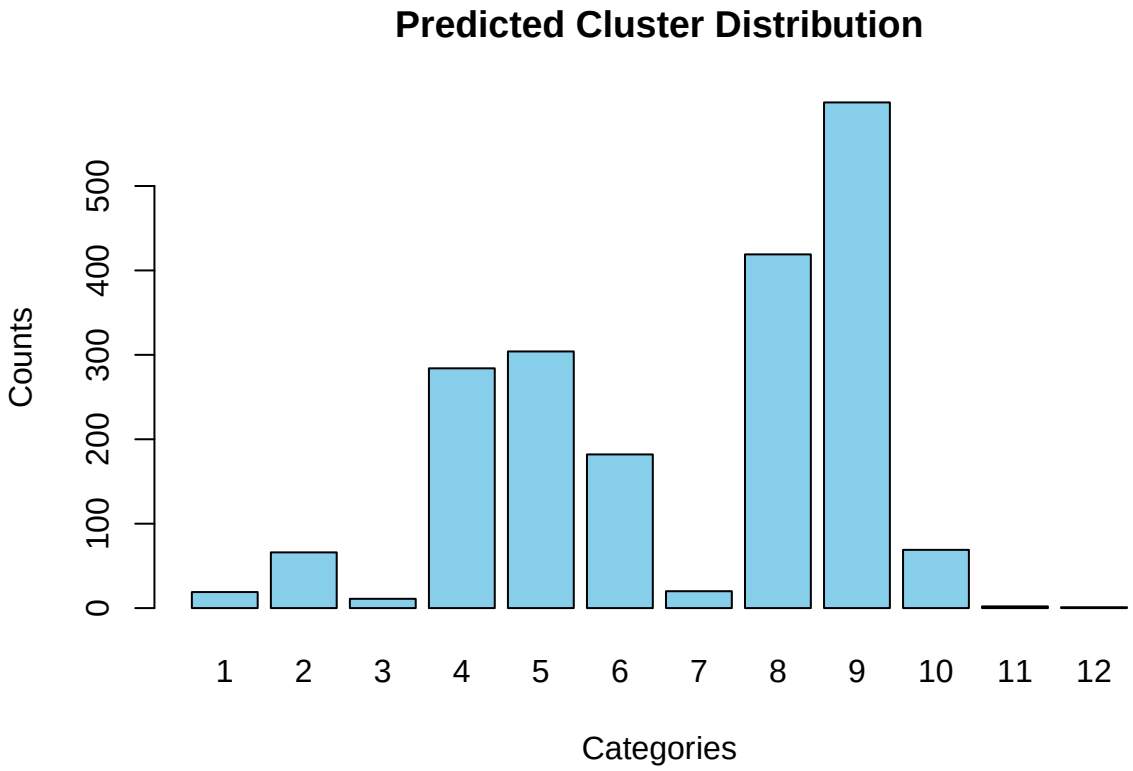
```
plot(condensed_data[, -c(1, 2)], col = cluster_labels, pch = 19, main = "K-Means Clustering")

# Add centroids to the plot
points(centroids, col = unique(cluster_labels), pch = 'X', cex = 2)
```

This is our preliminary K-Means clustering algorithm, performed on the training data. As we can see here, there is a very high dimensionality, which may possibly lead to issues with our model. Due to this aforementioned high dimensionality, we can take a look at the distributions of cluster labels to see if this model did a good job. If the distribution is similar, we can correctly assume the model performed adequately. To accomplish this, we use a bar chart that plots the counts of each of the labels.

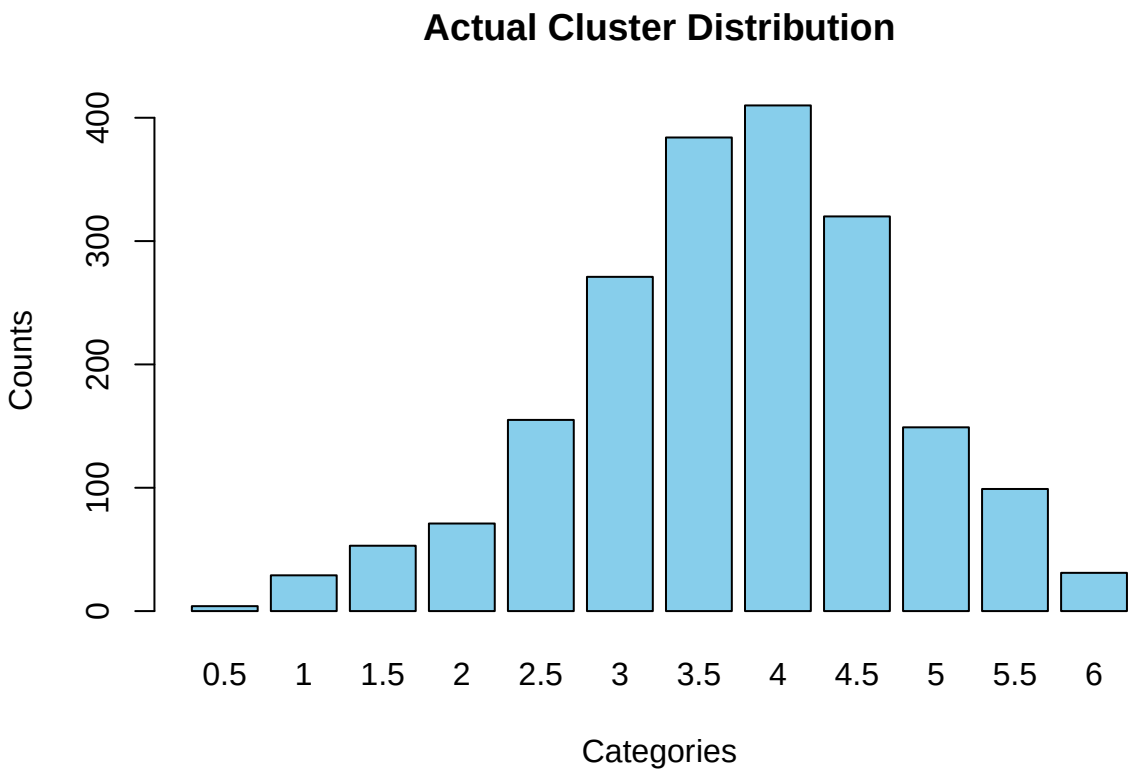
```
labels_count = table(cluster_labels)
labels_count
```

```
## cluster_labels
##   1   2   3   4   5   6   7   8   9  10  11  12
##  19  66  11 284 304 182  20 419 599  69   2   1
```

```
barplot(labels_count, names.arg = names(labels_count), col = "skyblue", main = "Predicted Cluster Distri
```

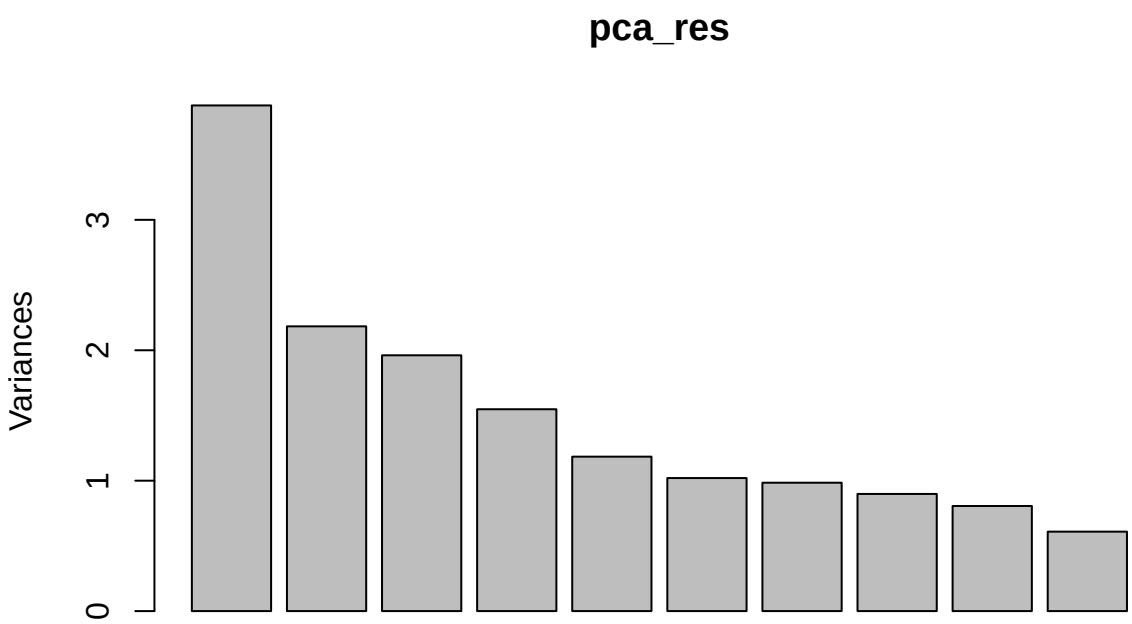



```
barplot(countsdf$counts, names.arg = countsdf$scores, col = "skyblue", main = "Actual Cluster Distributi
```



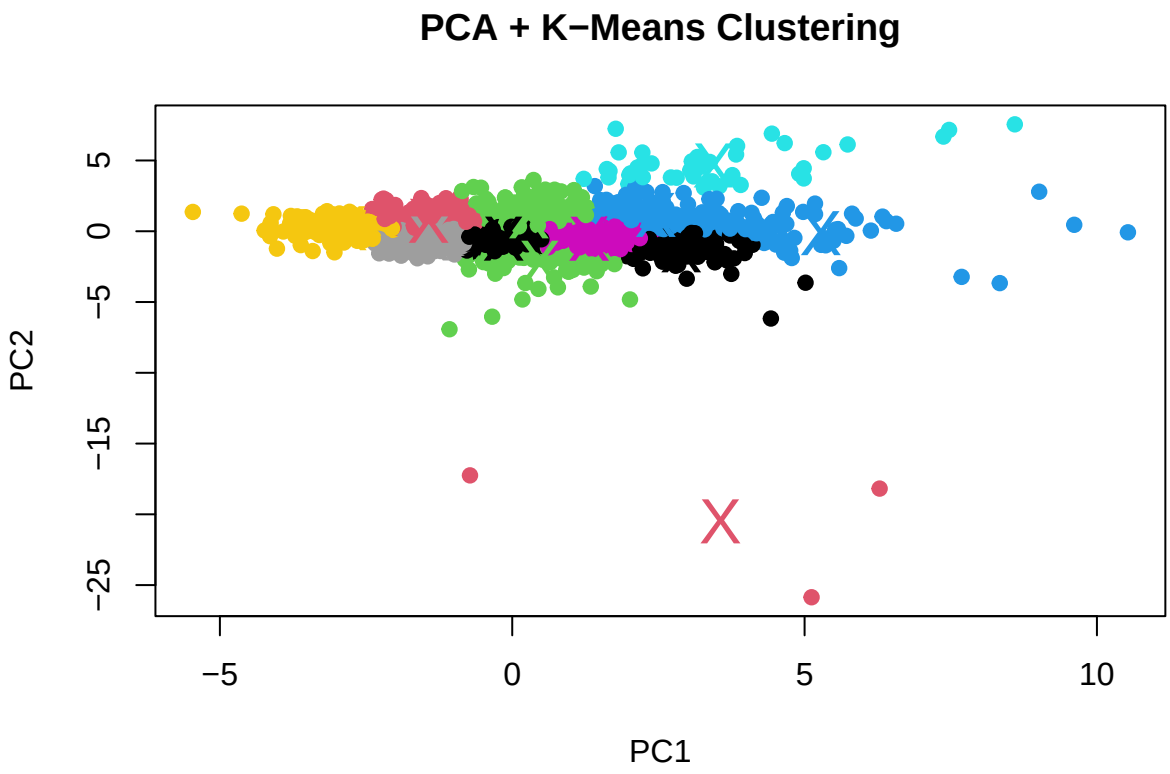
From these bar charts, we can see that the general shape is followed, but the predicted cluster distribution is definitely lacking a lot of information. Therefore, simply the K-Means model may not be enough to adequately represent this data. We saw an issue with dimensionality, as the data has seventeen predictors, so using Principal Component Analysis as a method of dimensionality reduction, followed by a fitted of a new K-Means model may lead to better results.

```
pca_res = prcomp(condensed_data[, -c(1, 2)], center=TRUE, scale.=TRUE)
plot(pca_res)
```

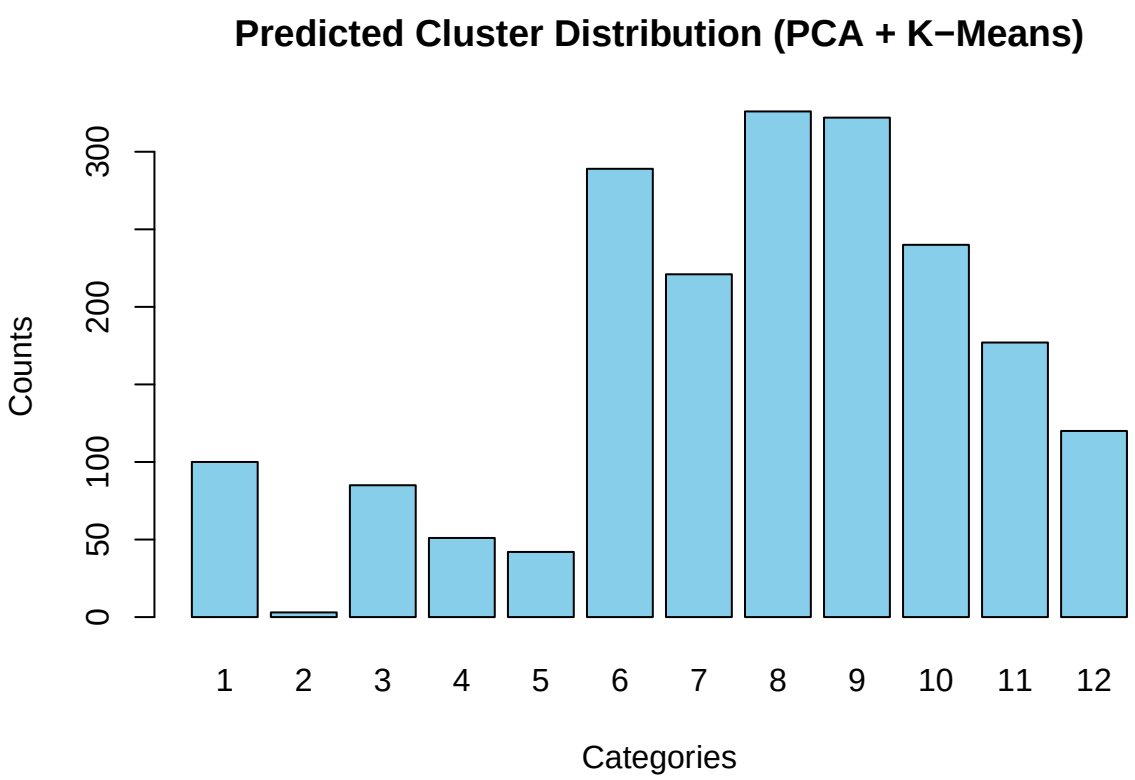


As we can see from the graph of principal components, the ones that explain the most variance in the data are approximately the first 2-4 components. For our purposes, we will retain the first 3 components as our new reduced dataset. We then fit a K-Means model with this data, utilizing 12 clusters as that is how many scores there are.

```
set.seed(138)
num_components = 2
reduced_data = pca_res$x[, 1:num_components]

kmeans_result = kmeans(reduced_data, centers = 12)

plot(reduced_data, col = kmeans_result$cluster, pch = 19, main = "PCA + K-Means Clustering")
points(kmeans_result$centers[, 1:num_components], col = 1:12, pch = 'X', cex = 2)
```




```
labels_count = table(kmeans_result$cluster)
barplot(labels_count, names.arg = names(labels_count), col = "skyblue", main = "Predicted Cluster Distri
```

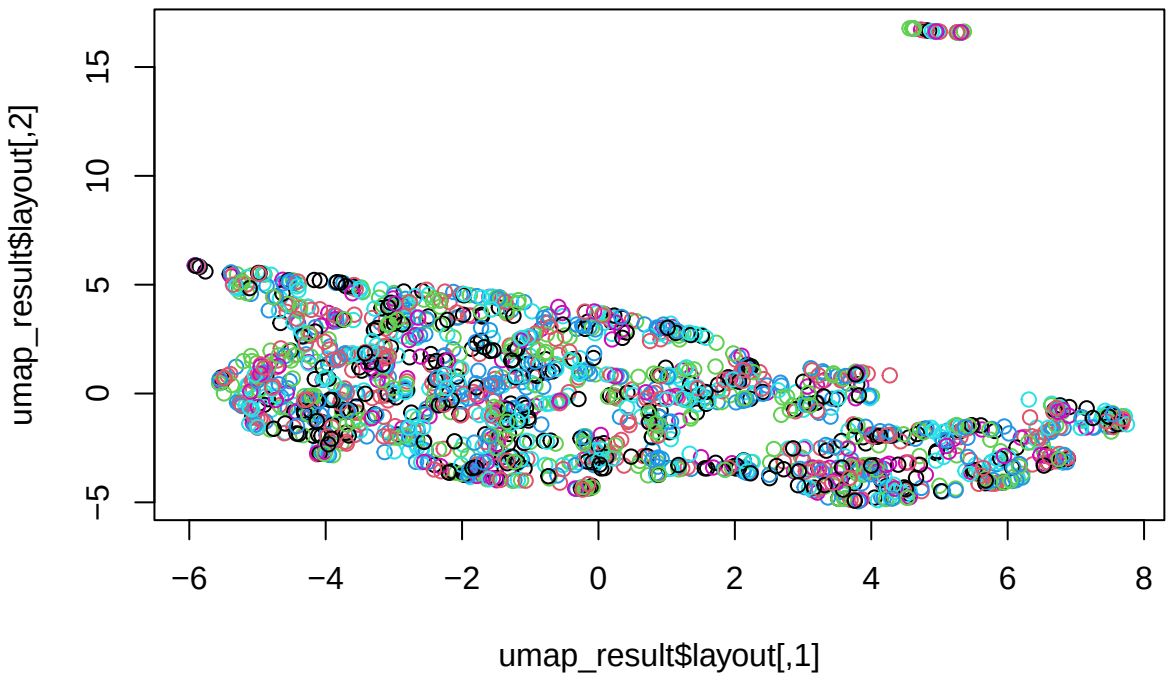


From this plot, we can see that the reduced data clusters very nicely, with a few exceptions towards the right-hand side of the plot. Furthermore, most of the centroids were accurately predicted using K-Means, which indicates a better model fit. The dimensionality reduction offered by Principal Component Analysis definitely helped, and allowed for a better fit.

Moreover, when we analyze the sets of c

Throughout both approaches of clustering, I opted to use 12 clusters as we found out in the training data that there are 12

```
library(umap)
umap_result <- umap(condensed_data[, -c(1, 2)], n_components = 2, n_neighbors = 15)
plot(umap_result$layout, col = scores)
```


umap_result$layout[,2]
15
10
5
0
−5
−6
−4
−2
0
2
4
6
8
umap_result$layout[,1]