Lecture 17 - Class Activity PCA

Bill Perry

# Lecture 17: Principal Component Analysis (PCA)

## What is PCA?

PCA (Principal Component Analysis) is a technique to: - Reduce the number of variables in your dataset - Find patterns in high-dimensional data - Create new uncorrelated variables (principal components) from correlated original variables - Visualize complex relationships in multivariate data

## When to Use PCA

Use PCA when you have: - **Multiple continuous variables** that may be correlated - **Too many variables** to analyze or visualize easily - **Need to reduce dimensionality** while retaining most information - **Want to explore patterns** in multivariate data

## Key Assumptions of PCA

1. **Linear relationships** between variables
2. **No extreme outliers** (can distort results)
3. **Variables should be correlated** (if not, PCA won’t reduce dimensions effectively)
4. **Adequate sample size** (generally n > 50)
5. **Consider standardization** when variables have different scales

|  |  |
| --- | --- |
|  | **Critical First Step**Always **standardize your data** when variables are measured on different scales. This prevents variables with larger values from dominating the analysis. |

# Part 1: Iris Data Analysis

## Data Overview

We’ll analyze the famous iris dataset with measurements from three species: - *Iris setosa* - *Iris versicolor* - *Iris virginica*

Each flower has 4 measurements: sepal length, sepal width, petal length, and petal width.

# Load the iris dataset from CSV
iris\_df <- read.csv("data/iris.csv") %>%
 clean\_names() %>%
 mutate(ind = row\_number()) %>%
 mutate(species\_ind = paste(species, ind, sep="\_"))

# View data structure
head(iris\_df)

 sepal\_length sepal\_width petal\_length petal\_width species ind species\_ind
1 5.1 3.5 1.4 0.2 setosa 1 setosa\_1
2 4.9 3.0 1.4 0.2 setosa 2 setosa\_2
3 4.7 3.2 1.3 0.2 setosa 3 setosa\_3
4 4.6 3.1 1.5 0.2 setosa 4 setosa\_4
5 5.0 3.6 1.4 0.2 setosa 5 setosa\_5
6 5.4 3.9 1.7 0.4 setosa 6 setosa\_6

# Get numeric values only for PCA
iris\_data\_df <- iris\_df %>%
 dplyr::select(sepal\_length, sepal\_width, petal\_length, petal\_width)

# Keep species info for later
iris\_species\_df <- iris\_df %>%
 dplyr::select(species, ind, species\_ind)

# Create long format for visualization
iris\_long\_df <- iris\_df %>%
 pivot\_longer(
 cols = c(sepal\_length, sepal\_width, petal\_length, petal\_width),
 names\_to = "variable",
 values\_to = "values"
 )

# Overview plot
overview\_plot <- iris\_long\_df %>%
 ggplot(aes(species, values, color = species)) +
 geom\_boxplot() +
 facet\_wrap(~variable, scales = "free") +
 labs(title = "Iris Measurements by Species",
 x = "Species",
 y = "Measurement Value") +
 theme\_minimal()

overview\_plot



## Step 1: Check PCA Assumptions

### Check Correlations

# Calculate correlation matrix
cor\_matrix <- cor(iris\_data\_df)

# Display correlation matrix
cor\_matrix

 sepal\_length sepal\_width petal\_length petal\_width
sepal\_length 1.0000000 -0.1175698 0.8717538 0.8179411
sepal\_width -0.1175698 1.0000000 -0.4284401 -0.3661259
petal\_length 0.8717538 -0.4284401 1.0000000 0.9628654
petal\_width 0.8179411 -0.3661259 0.9628654 1.0000000

# Visualize correlations
corrplot(cor\_matrix, method = "color", type = "upper",
 addCoef.col = "grey55", tl.cex = 0.8, number.cex = 0.8,
 title = "Correlation Matrix of Iris Variables",
 mar = c(0, 0, 2, 0))



### Check for Outliers

# Create boxplots to check for outliers
outlier\_plot <- iris\_data\_df %>%
 pivot\_longer(everything(), names\_to = "variable", values\_to = "value") %>%
 ggplot(aes(x = variable, y = value)) +
 geom\_boxplot() +
 labs(title = "Check for Outliers in Iris Variables",
 x = "Variable",
 y = "Value") +
 theme\_minimal() +
 theme(axis.text.x = element\_text(angle = 45, hjust = 1))

outlier\_plot



## Step 2: Standardize the Data

Since our variables have different scales (e.g., petal width ranges 0.1-2.5 while sepal length ranges 4-8), we need to standardize.

# Standardize the data (mean = 0, sd = 1)
iris\_scaled <- scale(iris\_data\_df)

# Convert back to data frame
iris\_scaled\_df <- as.data.frame(iris\_scaled)

# Check standardization worked
colMeans(iris\_scaled\_df)

 sepal\_length sepal\_width petal\_length petal\_width
-1.457168e-15 -1.638319e-15 -1.292300e-15 -5.543714e-16

apply(iris\_scaled\_df, 2, sd)

sepal\_length sepal\_width petal\_length petal\_width
 1 1 1 1

## Step 3: Perform PCA

# Run PCA on standardized data
iris\_pca\_model <- prcomp(iris\_scaled, center = FALSE, scale. = FALSE)

# View PCA summary
summary(iris\_pca\_model)

Importance of components:
 PC1 PC2 PC3 PC4
Standard deviation 1.7084 0.9560 0.38309 0.14393
Proportion of Variance 0.7296 0.2285 0.03669 0.00518
Cumulative Proportion 0.7296 0.9581 0.99482 1.00000

## Step 4: Extract and Understand Results

### Eigenvalues and Variance Explained

# Extract eigenvalues
eigenvalues <- iris\_pca\_model$sdev^2
prop\_variance <- eigenvalues / sum(eigenvalues)
cumsum\_variance <- cumsum(prop\_variance)

# Create summary table
pca\_summary\_df <- data.frame(
 Component = paste0("PC", 1:length(eigenvalues)),
 Eigenvalue = eigenvalues,
 Prop\_Variance = prop\_variance,
 Cumsum\_Variance = cumsum\_variance
)

pca\_summary\_df

 Component Eigenvalue Prop\_Variance Cumsum\_Variance
1 PC1 2.91849782 0.729624454 0.7296245
2 PC2 0.91403047 0.228507618 0.9581321
3 PC3 0.14675688 0.036689219 0.9948213
4 PC4 0.02071484 0.005178709 1.0000000

### Component Loadings

# Extract loadings (eigenvectors)
loadings\_df <- data.frame(
 Variable = rownames(iris\_pca\_model$rotation),
 PC1 = iris\_pca\_model$rotation[, 1],
 PC2 = iris\_pca\_model$rotation[, 2],
 PC3 = iris\_pca\_model$rotation[, 3],
 PC4 = iris\_pca\_model$rotation[, 4]
)

loadings\_df

 Variable PC1 PC2 PC3 PC4
sepal\_length sepal\_length 0.5210659 -0.37741762 0.7195664 0.2612863
sepal\_width sepal\_width -0.2693474 -0.92329566 -0.2443818 -0.1235096
petal\_length petal\_length 0.5804131 -0.02449161 -0.1421264 -0.8014492
petal\_width petal\_width 0.5648565 -0.06694199 -0.6342727 0.5235971

## Step 5: Determine Number of Components

### Scree Plot

# Create scree plot
scree\_data\_df <- data.frame(
 Component = factor(1:4),
 Variance = prop\_variance \* 100
)

scree\_plot <- ggplot(scree\_data\_df, aes(x = Component, y = Variance)) +
 geom\_col(fill = "steelblue") +
 geom\_line(aes(group = 1), size = 1) +
 geom\_point(size = 3) +
 labs(title = "Scree Plot - Variance Explained by Each Component",
 x = "Principal Component",
 y = "% of Variance Explained") +
 theme\_minimal()

Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
ℹ Please use `linewidth` instead.

scree\_plot



### Apply Decision Rules

# Eigenvalue > 1 rule
components\_to\_keep <- sum(eigenvalues > 1)
components\_to\_keep

[1] 1

# Components explaining at least 80% variance
components\_80\_percent <- which(cumsum\_variance >= 0.80)[1]
components\_80\_percent

[1] 2

## Step 6: Create PCA Scores

# Extract PCA scores
pca\_scores\_df <- data.frame(
 iris\_pca\_model$x,
 species = iris\_species\_df$species
)

# View first few rows
head(pca\_scores\_df)

 PC1 PC2 PC3 PC4 species
1 -2.257141 -0.4784238 0.12727962 0.024087508 setosa
2 -2.074013 0.6718827 0.23382552 0.102662845 setosa
3 -2.356335 0.3407664 -0.04405390 0.028282305 setosa
4 -2.291707 0.5953999 -0.09098530 -0.065735340 setosa
5 -2.381863 -0.6446757 -0.01568565 -0.035802870 setosa
6 -2.068701 -1.4842053 -0.02687825 0.006586116 setosa

## Step 7: Visualize Results

### PCA Scores Plot

# Create scores plot
scores\_plot <- ggplot(pca\_scores\_df, aes(x = PC1, y = PC2, color = species)) +
 geom\_point(size = 3, alpha = 0.7) +
 stat\_ellipse(level = 0.68, linetype = 2) +
 labs(title = "PCA Scores Plot",
 subtitle = paste0("PC1 explains ", round(prop\_variance[1]\*100, 1),
 "% of variance, PC2 explains ", round(prop\_variance[2]\*100, 1), "%"),
 x = paste0("PC1 (", round(prop\_variance[1]\*100, 1), "%)"),
 y = paste0("PC2 (", round(prop\_variance[2]\*100, 1), "%)"),
 color = "Species") +
 theme\_minimal() +
 scale\_color\_manual(values = c("#00AFBB", "#E7B800", "#FC4E07"))

scores\_plot



### Loading Plot

# Create loading plot
loading\_data\_df <- loadings\_df %>%
 dplyr::select(Variable, PC1, PC2)

loading\_plot <- ggplot(loading\_data\_df, aes(x = 0, y = 0)) +
 geom\_segment(aes(xend = PC1, yend = PC2),
 arrow = arrow(length = unit(0.3, "cm")),
 color = "red", size = 1) +
 geom\_text(aes(x = PC1 \* 1.1, y = PC2 \* 1.1, label = Variable),
 size = 4) +
 xlim(-1, 1) + ylim(-1, 1) +
 labs(title = "PCA Loading Plot",
 x = paste0("PC1 (", round(prop\_variance[1]\*100, 1), "%)"),
 y = paste0("PC2 (", round(prop\_variance[2]\*100, 1), "%)")) +
 theme\_minimal() +
 geom\_vline(xintercept = 0, linetype = "dashed", alpha = 0.5) +
 geom\_hline(yintercept = 0, linetype = "dashed", alpha = 0.5)

loading\_plot

Warning: Removed 1 row containing missing values or values outside the scale range
(`geom\_text()`).



### Combined Biplot

# Scale factor for arrows
arrow\_scale <- 3

# Create biplot manually
biplot\_plot <- ggplot(pca\_scores\_df, aes(x = PC1, y = PC2)) +
 # Add points for observations
 geom\_point(aes(color = species), size = 2, alpha = 0.6) +
 # Add arrows for variables
 geom\_segment(data = loading\_data\_df,
 aes(x = 0, y = 0,
 xend = PC1 \* arrow\_scale,
 yend = PC2 \* arrow\_scale),
 arrow = arrow(length = unit(0.3, "cm")),
 color = "black", size = 0.8) +
 # Add variable labels
 geom\_text(data = loading\_data\_df,
 aes(x = PC1 \* arrow\_scale \* 1.1,
 y = PC2 \* arrow\_scale \* 1.1,
 label = Variable),
 size = 3) +
 # Add ellipses
 stat\_ellipse(aes(color = species), level = 0.68, linetype = 2) +
 labs(title = "PCA Biplot - Iris Dataset",
 subtitle = "Points = Individual flowers, Arrows = Original variables",
 x = paste0("PC1 (", round(prop\_variance[1]\*100, 1), "%)"),
 y = paste0("PC2 (", round(prop\_variance[2]\*100, 1), "%)"),
 color = "Species") +
 theme\_minimal() +
 scale\_color\_manual(values = c("#00AFBB", "#E7B800", "#FC4E07"))

biplot\_plot



## Step 8: Interpret Results

# PC1 interpretation
pc1\_loadings <- iris\_pca\_model$rotation[, 1]
pc1\_loadings

sepal\_length sepal\_width petal\_length petal\_width
 0.5210659 -0.2693474 0.5804131 0.5648565

# PC2 interpretation
pc2\_loadings <- iris\_pca\_model$rotation[, 2]
pc2\_loadings

sepal\_length sepal\_width petal\_length petal\_width
 -0.37741762 -0.92329566 -0.02449161 -0.06694199

# Variance explained summary
total\_variance\_2pc <- sum(prop\_variance[1:2])
total\_variance\_2pc

[1] 0.9581321

# Summary Checklist for PCA

When conducting PCA, always follow these steps:

|  |  |
| --- | --- |
|  | **PCA Checklist**1. **Explore your data** - check distributions and relationships
2. **Check correlations** - PCA works best with correlated variables
3. **Check for outliers** - they can distort results
4. **Standardize if needed** - essential when variables have different scales
5. **Run PCA** - extract components
6. **Determine number of components** - use scree plot and variance explained
7. **Interpret loadings** - understand what each component represents
8. **Visualize results** - create scores plots and biplots
9. **Validate interpretation** - ensure it makes biological sense
 |

## Key Points to Remember

* **PCA finds new variables** (components) that are linear combinations of original variables
* **Components are uncorrelated** and ordered by variance explained
* **Standardization is crucial** when variables have different units/scales
* **First few components** usually capture most variation
* **Loadings show** how original variables contribute to components
* **Scores show** where observations fall in the new component space

|  |  |
| --- | --- |
|  | **Key Points from PCA Analysis**1. **Check assumptions first** - especially correlations and outliers
2. **Standardize when necessary** - prevents scale effects from dominating
3. **Use multiple criteria** to decide number of components (scree plot, eigenvalue > 1, variance explained)
4. **Interpret components** based on loadings - what do they represent biologically?
5. **Visualize in 2D** using first two components if they explain sufficient variance
6. **PCA is exploratory** - use it to understand patterns, not for hypothesis testing
7. **Document your choices** - why you kept certain components, how you interpreted them

Remember: PCA is a dimension reduction technique - the goal is to simplify complex data while retaining the important patterns! |