kusum_khatri_15

kusum_khatri_15

2024-05-31

R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see http://rmarkdown.rstudio.com.

When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

```{# qn 7 aq <- airquality aq # Calculate mean of the wind by month using apply function mean_wind <- tapply(aq$Temp$, $aq$Month, mean, na.rm = T) mean_wind

## Calculate the standard deviation of wind by month using tapply function

sd_wind <- tapply(aq$Temp$, $aq$Month, sd, na.rm = T) sd_wind # Creating a data frame to show the output mean_sd_table <- data.frame( Mean_wind = mean_wind, SD_wind = sd_wind )

## Display the table

print(mean_sd_table)

## Test of normality

## Perform Shapiro-Wilk test for each month

result <- tapply(aq$Wind$, $airquality$Month, shapiro.test) #ks test

## Print the results

print(result) #Perform goodness-of-fit test on Wind variable by Month variable to check #if the variances of mpg are equal or not on am variable categories
```

Convert Month to a factor

airquality$Month < -factor(airquality$Month)

Perform Bartlett's test for homogeneity of variances #levene

bartlett_result <- bartlett.test(Temp ~ Month, data = airquality)

Print Bartlett's test result

print(bartlett_result) #Discuss which one-way ANOVA must be used to compare "Wind" #variable by "Month" variable categories based on the results obtained above

#In the above scenario, Bartlett's test indicates that the variances of the "Wind" variable are approximately equal across different months. # Therefore, we can use the standard one-way ANOVA. #Fit the best one-way ANOVA for this data now and interpret the results carefully # Load the airquality dataset if not already loaded

data("airquality")

Fit one-way ANOVA model

anova_model <- aov(Wind ~ Month, data = airquality)

Summary of the ANOVA model

summary(anova_model) #Fit the most-appropriate post-hoc test if the ANOVA is statistically significant #and interpret the result carefully

Convert Month to a factor

airquality$Month < -factor(airquality$Month)

Fit one-way ANOVA model

anova_model <- aov(Wind ~ Month, data = airquality)

Perform Tukey's HSD test

tukey_result <- TukeyHSD(anova_model)

Print the Tukey HSD test result

print(tukey_result)

} { ir <- iris ir library(factoextra) # when k = 2 kmeans_result <- kmeans(sd.ir, nstart = 2) fviz_cluster(kmeans_result, data = sd.data, ellipse.type = “norm”, geom = “point”, stand = FALSE, main = “K-means Clustering (fviz_cluster)”)

kmeans_result2 <- kmeans(sd.ir, nstart = 3) fviz_cluster(kmeans_result, data = sd.data, ellipse.type = “norm”, geom = “point”, stand = FALSE, main = “K-means Clustering (fviz_cluster)”)

Perform k-means clustering

set.seed(12) kmeans_result3 <- kmeans(sd.data, centers = optimal_k, nstart = 20) fviz_cluster(kmeans_result, data = sd.data, ellipse.type = “norm”, geom = “point”, stand = FALSE, main = “K-means Clustering (fviz_cluster)”)

Get summary of the k-means clustering

print(kmeans_result3)

Interpret the results

cat(“Cluster Centers:”) print(kmeans_result3$centers)

cat(“Sizes:”) print(kmeans_result3$size)

cat(“-cluster sum of squares:”) print(kmeans_result3$withinss)

cat(“within-cluster sum of squares:”) print(kmeans_result3$tot.withinss)

cat(“-cluster sum of squares:”) print(kmeans_result3$betweenss)

Plot using base R plot

par(mfrow = c(1, 1)) # Reset to 1 plot per row plot(sd.data, col = kmeans_result$cluster, pch = 16, main = “K-means Clustering”)

Plot using cluster package

clusplot(sd.data, kmeans_result$cluster, color = TRUE, shade = TRUE, labels = 2, lines = 0, main = “K-means Clustering (clusplot)”)

so this varibale when = 3 this cluster will be created as shown on the diagram which to get

kmean cluster perfectly divide all the value and varibale and divide into different cluster as shown in diagram

k mean cluster always caluclate value by the centroid and help to get the result

} {

#8 library(ggplot2) flowerscale <- iris flowerscale

Define the number of samples and variables

num_samples <- 150 num_variables <- 4

Generate random data matrix

iris <- matrix(runif(num_samples*num_variables), ncol=num_variables)

Set row names

rownames(iris) <- paste0("Row", 1:150)

Set column names

colnames(iris) <- paste0("Column", 1:4)

Display the first few rows of the random data

head(iris)

Compute the correlation matrix

corr_matrix <- cor(iris) ggcorrplot(corr_matrix)

Perform Principal Component Analysis (PCA)

pca <- prcomp(iris, scale = TRUE)

Display PCA results

pca

Extract the names of the principal components

names(pca)

Summary of PCA analysis

summary(pca)

Compute the proportion of variance explained by each principal component

pca.var <- pca$sdev^2 propve <- pca.var / sum(pca.var) propve

Plot variance explained for each principal component

plot(propve, xlab = “Principal Component”, ylab = “Proportion of Variance Explained”, ylim = c(0, 1), type = “b”, main = “Scree Plot”)

Plot the cumulative proportion of variance explained

plot(cumsum(propve), xlab = “Principal Component”, ylab = “Cumulative Proportion of Variance Explained”, ylim = c(0, 1), type = “b”)

Determine the number of principal components needed to explain 90% variance

which(cumsum(propve) >= 0.9)[1]

Load the psych library for factor analysis

library(psych)

Perform Factor Analysis with Varimax rotation

pca_varimax <- principal(iris, nfactors=num_variables, rotate="varimax")

Display Factor Analysis results

pca_varimax

Compute the distance matrix

distance_matrix <- dist(iris)

Plot the eigenvalues of PCA

fviz_eig(pca, addlabels = TRUE)

Load necessary libraries for visualization

library(ggplot2) library(reshape2)

Graph of the variables

fviz_pca_var(pca, col.var = "black")

fviz_cos2(pca, choice = "var", axes = 1:4)

Create PCA plot

pca_plot <- ggplot(pca_df, aes(x=PC1, y=PC2)) + geom_point() + labs(title="PCA")

Print explained variance ratios of PCA

print("Explained Variance Ratios of PCA:") print(summary(pca)$importance[2,])

Print explained variance ratios of Factor Analysis

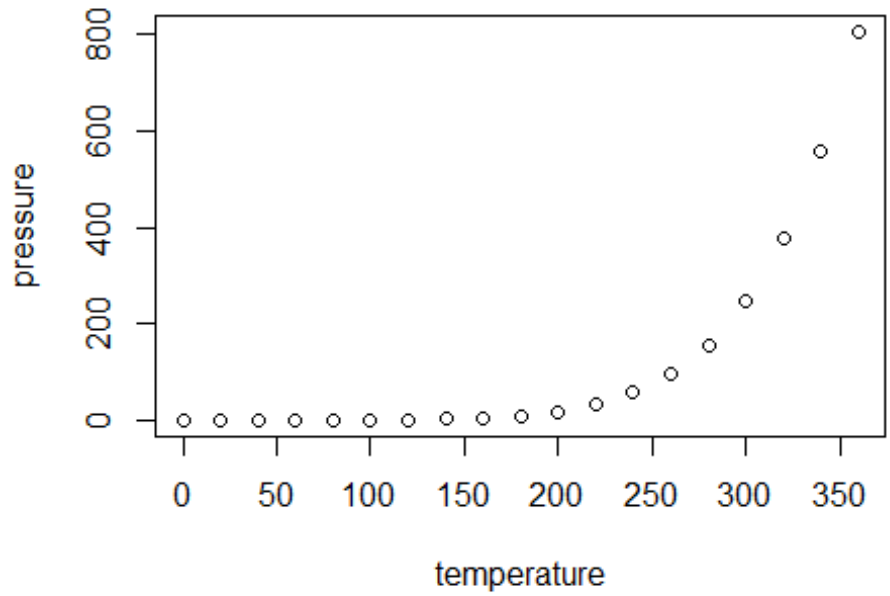
print("Explained Variance Ratios of Factor Analysis:") print(pca_varimax$values)

} {}

```
```

Including Plots

You can also embed plots, for example:



Note that the `echo = FALSE` parameter was added to the code chunk to prevent printing of the R code that generated the plot.