Comprehensive Study Guide for Statistical Unsupervised Learning

March 9, 2025

Contents

1 Introduction to Unsupervised Learning

Unsupervised learning methods infer structures from unlabeled data by simplifying data structures and discovering patterns. Unlike supervised learning, it has no predefined labels.

2 Cluster Analysis

2.1 Similarity and Dissimilarity Matrices

A **similarity matrix (n x n)** quantifies similarity between pairs of observations (rows), whereas an ordinary **correlation matrix (p x p)** measures relationships between variables.

2.2 Distance Measures

Euclidean Distance: Straight-line distance between two points.

$$d_E(x, y) = \sqrt{\sum_{i=1}^{p} (x_i - y_i)^2}$$

Manhattan Distance: Sum of absolute differences.

$$d = \sum_{i=1}^{p} |x_i - y_i|$$

3 Clustering Methods

3.1 Hierarchical Clustering

Agglomeratively merges clusters iteratively based on chosen linkage criteria (single, complete, average, Ward's).

Ward's Method: Minimizes within-cluster variance increase (WSS), maximizing between-cluster variance (BSS).

R Example:

```
dist_matrix <- dist(data)
hc <- hclust(dist_matrix, method="ward.D2")
plot(hc)
```

3.2 K-means Clustering

Partitions data into k clusters by minimizing within-cluster variance.

Optimal clusters (Elbow Method): Plot WSS vs. k, choose k at "elbow."

R Implementation:

```
kmeans_result <- kmeans(data, centers=3, nstart=25)
```

3.3 Density-Based Clustering (DBSCAN)

Clusters are density-based; identifies noise points as well.

Parameters: *eps* (radius), *minPts* (min points per cluster).

R Implementation:

```
library(dbscan)
db <- dbscan(data, eps=0.5, minPts=5)
kNNdistplot(data, k=4)
```

3.4 Gaussian Mixture Models (GMM)

Probabilistically clusters data based on Gaussian distributions. Uses Expectation-Maximization algorithm.

R Implementation:

```
library(mclust)
gmm <- Mclust(data)
summary(gmm)
```

4 Dimensionality Reduction Techniques

4.1 Principal Component Analysis (PCA)

Converts correlated variables into uncorrelated principal components, maximizing explained variance.

Key concepts: Eigenvectors, eigenvalues, loadings, scree plot.

R Implementation:

```
pca <- prcomp(data, scale=TRUE)
summary(pca)
biplot(pca)
```

4.2 t-SNE (t-distributed Stochastic Neighbor Embedding)

Preserves local structures for high-dimensional data visualization using probability distributions.

Perplexity: Balances local and global structure.

R Implementation:

```
library(Rtsne)
tsne <- Rtsne(data, dims=2, perplexity=30)
plot(tsne$Y)
```

4.3 Independent Component Analysis (ICA)

Separates multivariate signals into independent components by maximizing non-Gaussianity.

Cocktail Party analogy: Separating mixed voices.

R Implementation:

```
library(fastICA)
ica_result <- fastICA(data, n.comp=3)
pairs(ica_result$S)
```

5 Association Rules

5.1 Apriori Algorithm

Finds frequent itemsets and derives association rules based on support, confidence, and lift.

Apriori property: Subsets of frequent itemsets are also frequent, reduces candidate sets.

R Implementation:

```
library(arules)
data("Groceries")
rules <- apriori(Groceries, parameter=list(supp=0.01, conf=0.5))
inspect(sort(rules, by="lift"))
```

6 Anomaly Detection

6.1 Isolation Forest

Identifies anomalies by isolating points with random partitions, using path length as anomaly score.

R Implementation:

```
library(isotree)
iso_forest <- isolation.forest(data, ntrees=100)
scores <- predict(iso_forest, data)
```

6.2 One-Class SVM

Uses hyperplanes in high-dimensional spaces to separate anomalies from normal data.

R Implementation:

```
library(e1071)
svm_model <- svm(data, type='one-classification', nu=0.1, kernel='radial')
predictions <- predict(svm_model, data)
```

7 Self-Organizing Maps (SOM)

Maps data to a two-dimensional grid preserving topological relationships.

Key Concepts:

- **Best Matching Unit (BMU)**: Closest neuron to input data.
- **Training Steps**: Competition (find BMU), Cooperation (adjust neighbors), Adaptation (iteratively update).

R Implementation:

```
library(kohonen)
som_grid <- somgrid(xdim=5, ydim=5, topo="hexagonal")
som_model <- som(data_scaled, grid=som_grid, rlen=100, alpha=c(0.05,0.01))
plot(som_model, type="property", property=getCodes(som_model)[,1])
```

8 Applications and Use Cases

- Market Segmentation
- Fraud Detection
- Bioinformatics
- Network Security
- Image and Speech Analysis
- Financial Portfolio Optimization