Rensselaer

why not change the world?®

More Supervised & Unsupervised Methods

Ahmed Eleish

Data Analytics ITWS-4600/6600 CSCI-4960 MGMT 4600/6600

November 6th 2026



Tetherless World Constellation
Rensselaer Polytechnic Institute

Contents

- Classification
  - XGBoost
- Clustering
  - DBSCAN
- Dimension Reduction
  - Factor Analysis

More ML methods

https://rpi.box.com/s/x6627gn22fjk8mgykdkecq44bo9xcpmy

eXtreme Gradient Boosting (XGBoost)

- Similar to Random Forest, XGBoost is based on decision trees.
- Many trees are built, one at a time, each learning from the errors of previous trees.
- Uses gradients of the loss function to minimize errors.
- Utilizes boosting: combining multiple weak classifiers sequentially to form a strong classifier (an ensemble)
- The final prediction is the weighted sum of the predictions from all the trees.

Evolution of Tree Algorithms



https://www.enjoyalgorithms.com/blog/xg-boost-algorithm-in-ml

Algorithm

- objective at boosting step (t) is:

$$\mathcal{L}^{(t)} = \sum_{i=1}^{n} \ell \left(y_i, \hat{y}_i^{(t-1)} + f_t(x_i) \right) + \Omega(f_t)$$

where ℓ measures prediction error, and f_t is the new tree. The regularization term is:

$$\Omega(f_t) = \gamma T + \frac{\lambda}{2} \sum_{j=1}^{T} w_j^2$$

where $\boldsymbol{T}$ is the number of leaves and $\boldsymbol{w_j}$ is a leaf's score. γ penalizes adding leaves, while λ shrinks leaf scores.

XGBoost Algorithm - Training

- **Initialize predictions.** (constant values e.g. mean of target var for regresion, 0.5 for classification)

For m = 1,2,3, …, M:

1. Calculate 1st and 2nd Derivatives of the loss with respect to the current prediction

$$g_i = \frac{\partial \ell(y_i, \hat{y}_i)}{\partial \hat{y}_i}, h_i = \frac{\partial^2 \ell(y_i, \hat{y}_i)}{\partial \hat{y}_i^2}.$$

2. Grow a tree to improve predictions.
candidate splits are scored using the derivatives, rewarding splits that reduce the loss.

3. Assign a score to each leaf.
Each leaf gets a value that adjusts predictions for examples routed there. In the basic case, its optimal score is:

4. Update predictions.
Add the new tree's scores to the current predictions, scaled by the learning rate η:

$$\hat{y}_i \leftarrow \hat{y}_i + \eta f_t(x_i).$$

XGBoost Algorithm - Prediction

- Given a new data point:
  - Traverse each tree to get a score (prediction)
  - The weighted sum of scores from all trees becomes the final prediction (regression) or is converted to probabilities for each class label and the most likely is selected (classification).

eXtreme Gradient Boosting (XGBoost)

- Pros:
  - Scalability, support for parallel processing and efficient data structures.
  - Customizability with multiple hyperparameters.
  - Can capture complex non-linear relationships.
- Cons:
  - Limited interpretability.
  - Can be computationally expensive.
  - Requires tuning, may overfit to smaller datasets.

Factor Analysis

- Identifies latent variables (factors) between correlated variables.
- Can be used for dimension reduction, hypothesis testing, and feature selection.
- Commonly used in psychology, marketing, biology and others.
- **Trade-off:** results depend on modeling choices, and factors can require careful interpretation.

Algorithm

1. Standardize data (center + scale by SD)
2. Compute correlation matrix *R*
3. Determine number of factors
4. Extract initial factors
5. Rotate factors
6. Interpret factors

Factor Analysis - Algorithm

1. **Standardize data** (center + scale by SD)
2. **Compute the correlation matrix:**
   Factor analysis requires variables to have shared variation.
3. **Choose the number of factors**
   Combine subject-matter knowledge with tools such as parallel analysis and the scree plot.
4. **Compute the factors**
   Use an extraction method such as maximum likelihood or principal axis factoring to estimate factor loadings and unique variances.
5. **Rotate for easier interpretation**
   Rotation can make the loading pattern clearer.
6. **Interpret and evaluate**
   Examine which variables load on each factor, along with model fit and residual correlations. Name factors based on the variables and the study context.

Factor Analysis

- Factor analysis models each observed data point as a combination of a few **latent factors** plus variable-specific noise:

$$xi = \Lambda f i + \epsilon i$$

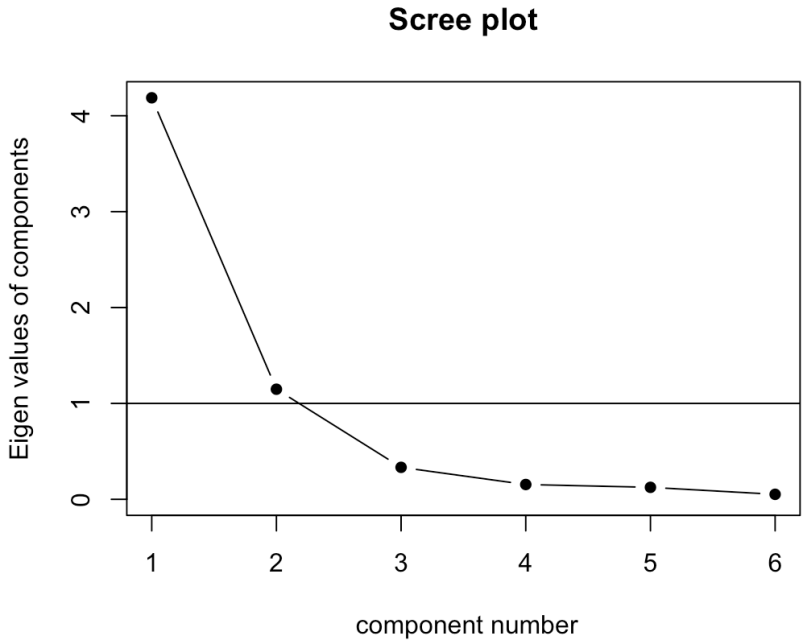
- The factor model says the observed covariance matrix is:

$$\hat{R} = \hat{\Lambda}\hat{\Phi}\hat{\Lambda}^{\mathrm{T}} + \hat{\Psi}$$

wh ere, $\hat{\Lambda}$ contains estimated loadings, $\hat{\Phi}$ contains factor correlations (the identity matrix if factors are uncorrelated), and $\hat{\Psi}$ contains unique variances.

Choosing the number of factors

1. Plot the sorted eigenvalues of the correlation matric
2. Look for the elbow
3. Choose the number of features at the elbow



Factor Analysis

Pros

- Reduces complexity and summarizes correlated features with fewer latent dimensions.
- Supports theory development and testing.

Cons

- Requires judgment in model choices which affect the outcomes of the analysis.
- Interpretation of factors can be subjective, and different solutions may be plausible.
- Does not establish causation!

Density-Based Spatial Clustering of Applications with Noise (DBSCAN)

- A density-based clustering algorithm -> groups points in dense regions.
- It identifies core points with enough nearby neighbors, then expands clusters through connected dense regions.
- Two key parameters:
    - epsilon (ε): maximum distance between points to be considered part of the same cluster.
    - minPts: the minimum number of points required to form a dense region.
- Especially effective for datasets containing noise and clusters of varying densities.

Sadeghi, B. (2025). Clustering in geo-data science: Navigating uncertainty to select the most reliable method. *Ore Geology Reviews*, 106591. https://doi.org/10.1016/j.oregeorev.2025.106591

Density-Based Spatial Clustering of Applications with Noise (DBSCAN)

Points are categorized as:

- **Core:** has at least *MinPts* points in its ε-neighborhood.
- **Border:** is near a core point but does not itself meet the *MinPts* threshold.
- **Noise:** is not assigned to a cluster under these rules.

- Unlike methods such as k-means, DBSCAN does not minimize a single general-purpose objective function.
- It forms clusters based on density reachability.

Sadeghi, B. (2025). Clustering in geo-data science: Navigating uncertainty to select the most reliable method. *Ore Geology Reviews*, 106591. https://doi.org/10.1016/j.oregeorev.2025.106591

Algorithm

1. Initialize (mark points as unvisited)
2. For each unvisited point *P:*
    2.a. Mark *P* as visited
    2.b. Find all neighbors *N* of *P* within ε distance
    2.c. If |N| < minPts: mark *P* as noise
    2.d. Else: create new cluster *C* and add *P* to it
3. Expand cluster *C* from *P,* for each point *Q* in neighborhood *N*
    3.a. Add Q to cluster C
    3.b. If *Q* is a core point, expand cluster *C: Repeat 3 until no more core points reachable*
4. Repeat 2-3 until all points are visited

DBSCAN

- Pros:
  - The number of clusters is not predefined.
  - Find arbitrary, non-spherical clusters.
  - Robust to outlier and good at identifying noise.
  - Suitable for geo-spatial data.

- Cons:
  - Not suitable for high-dimensional data where distance becomes less informative.
  - Assumes clusters of similar density.
  - Sensitivity to parameter choices.

Thanks!