**🧭 Unsupervised Analysis Workflow for Strategy Discovery**

**Step 1: Define Objective**

**Goal**: Discover distinct patterns of gaze behavior that may correspond to latent cognitive strategies (e.g., planner, explorer, checker) without using predefined labels.

**Step 2: Prepare the Data**

* ✅ Collect or import your **eye-tracking feature dataset** (e.g., 150 samples × 10 features).
* 🔍 Select a subset of **interpretable features** commonly linked to strategy:
  + Fixation duration (mean, max)
  + Entropy of transitions
  + Scanpath length
  + AOI revisits
  + Time to first fixation on target
  + Dwell time
* ❗ Handle missing values, smooth noise, and ensure consistent scale (see Step 3).

**Step 3: Preprocess the Features**

* ⚖️ **Normalize** features (z-score or min-max scaling)
* 🧹 **Clean**: remove extreme outliers or use winsorization
* 📊 (Optional) **Reduce dimensionality** using PCA before clustering

**Step 4: Run Unsupervised Clustering**

Apply 1–2 clustering methods:

* 🔷 **K-Means** if you expect ~3 strategies and want simplicity
* 🌐 **HDBSCAN** if you want robust, noise-tolerant, shape-flexible clusters
* 🌀 Optionally test **GMM** for probabilistic cluster assignments

**Step 5: Visualize the Clusters**

* Use **t-SNE or UMAP** to project high-dimensional data into 2D
* Plot points colored by cluster ID
* Observe cluster separation, density, and overlap

**Step 6: Interpret Clusters**

* Create a **cluster summary table**: mean/median of key metrics by cluster
* Identify meaningful **strategy labels** based on feature profiles:
  + Low entropy, short paths → likely **planner**
  + High entropy, many revisits → likely **explorer**
  + Late target fixations, long dwell → likely **checker**
* Validate against theoretical expectations (Hegarty, Lohman)

**Step 7: Iterate or Compare**

* Adjust clustering parameters (e.g., min\_cluster\_size, k)
* Compare results with:
  + Rule-based labels (if available)
  + External variables like task type or performance

**Step 8: Export & Use Results**

* Save cluster IDs as new labels
* Use them to:
  + Train a **supervised model** (e.g., Random Forest)
  + Analyze performance differences across strategies
  + Visualize scanpaths per cluster

**🔁 Optional: Triangulate with Qualitative Data**

* Overlay heatmaps or scanpaths from different clusters
* Cross-check with task logs, strategy interviews, or expert judgments

**🔍 Workflow: Explore Many Metrics → Narrow Down Meaningfully**

**✅ Step 1: Extract Broad Feature Set**

Start with **20–40 metrics** across these categories:

* **Fixation metrics** (e.g., duration, count, spatial density)
* **Saccade metrics** (e.g., amplitude, frequency)
* **Scanpath metrics** (e.g., length, entropy, direction)
* **AOI metrics** (e.g., TFF, revisits, dwell time)
* **Pupil & blink metrics** (optional: pupil dilation, blink rate)

**🧪 Step 2: Apply Feature Reduction Techniques**

**1. Correlation Matrix**

Identify redundant features (e.g., fixation count and duration are often collinear)

python

CopyEdit

import seaborn as sns

import matplotlib.pyplot as plt

corr = df.corr()

sns.heatmap(corr, cmap='coolwarm')

* Drop one of any pair with correlation > 0.9

**2. Unsupervised Filtering**

Useful if you don't have labels yet

* **Variance Threshold**: Drop low-variance features
* **PCA / UMAP**: Project into lower dimensions to explore redundancy

**3. Supervised Filtering**

Once labels are available (e.g., planner, explorer)

**a. Feature Importance (Random Forest)**

python

CopyEdit

from sklearn.ensemble import RandomForestClassifier

model = RandomForestClassifier()

model.fit(X, y)

importance = model.feature\_importances\_

* Select top 10–15 metrics

**b. Lasso (L1-Regularized Regression)**

* Forces some coefficients to zero—automatically selects a subset

**4. Recursive Feature Elimination (RFE)**

Systematically removes least important features using a model like RF or SVM.

python

CopyEdit

from sklearn.feature\_selection import RFE

rfe = RFE(RandomForestClassifier(), n\_features\_to\_select=10)

rfe.fit(X, y)

**5. Clustering-Specific Relevance**

When using unsupervised clustering, try:

* **Silhouette analysis** with subsets of features
* **Cluster stability analysis** when features are added/removed

**🧠 Tip: Retain Theory-Relevant Features**

Even if some features appear “weak” statistically, keep those:

* Tied directly to cognitive theory (e.g., entropy → strategy exploration)
* With high interpretability or prior empirical value

**📌 Summary Table**

| **Method** | **When to Use** | **Goal** |
| --- | --- | --- |
| Correlation Filtering | Before any modeling | Remove redundancy |
| PCA / UMAP | No labels yet | Explore feature overlap |
| Random Forest Importance | After label creation | Identify predictive features |
| RFE or LASSO | Model-driven feature selection | Build compact models |
| Theoretical Selection | Always | Preserve interpretability |