BA820 – Project M4

Cover Page

- **Project Title:** Interpretable Palette Archetypes in Bob Ross Paintings
- **Section and Team Number:** B1 Team 3
- **Student Name:** Irene Tang

1. Refined Problem Statement & Focus

Business question unchanged from M2: **Do Bob Ross paintings form a small number of interpretable palette archetypes based on binary paint usage?**

What changed in M4:

The framing and success criteria changed in M4: rather than expecting several balanced archetypes, I treat the dominant cluster as a baseline "standard palette" and evaluate whether deviations are meaningful through robustness checks, paint-combination signatures, and external validation. M2 and M3 already showed one dominant "standard palette" (Cluster 4, 385 paintings, around 95.5%) along with a few small clusters (10, 5, 2, 1). M4 refines this in the following:

- Linking clusters to metadata and title themes to check whether rare clusters align with season, episode, or theme.
- Comparing linkage methods and rule parameters for rigor.
- Comparing palette clusters to title-based clusters (CountVectorizer + KMeans + cosine similarity) to see if text and palette structure align.

Assumptions validated:

The extreme imbalance appears to be structural rather than a coding artifact, which is supported by linkage-comparison and threshold experiments in M4. Assumption that small clusters can be "interpretable archetypes" is tested by checking whether they concentrate in certain seasons or title themes in M4. The crosstabs show the dominant cluster spans all seasons/themes, while small clusters have no strong concentration. So, I describe their distribution rather than claiming a one-to-one theme mapping.

2. EDA & Preprocessing: Updates

No new EDA or preprocessing was added in M4. I reuse the M2/M3 pipeline: each painting is represented by 18 binary paint indicators and palette similarity is treated as set overlap, so Jaccard distance remains the primary metric for clustering. For association-rule mining, Titanium_White (global usage ≥ 0.95) is removed to avoid trivial, near-constant items dominating the resulting rules and to improve interpretability of co-usage patterns.

These choices align with the project goal of treating palettes as sets, and they make both clustering and co-occurrence rule mining interpretable under the baseline–deviation framing. The only M4-specific addition is a lightweight title_theme label (for example, landscape, water, season_time, objects_view.) created via keyword matching on painting titles purely to support interpretation of cluster × theme crosstabs; it does not change the underlying clustering or rule-mining inputs.

3. Analysis & Experiments

Building on M3, I continue the same two unsupervised methods from earlier milestones because the data is binary paint usage, and the goal is to recover interpretable palette structure rather than

predict an outcome. M3 established that Jaccard-based hierarchical clustering consistently produces a baseline-heavy solution and that association rules help interpret deviations beyond what clustering alone can explain. In M4, I treat the dominant-cluster outcome as a structural feature rather than a failure to find balanced archetypes. And I focus on stress-testing whether the baseline–deviation pattern is an artifact of method choices, translating deviations into concrete paint-combination signatures, and checking whether deviations align with external signals such as season/episode metadata and title text structure.

3.1 Clustering robustness

In M4, I would like to know whether the dominant baseline is an artifact. The immediate concern from M3 is that a ~95% cluster could look "unreasonable" if the method is forcing everything together (Plot 1). Therefore, the goal is to test whether the baseline-deviation structure is a real property of the data or a byproduct of a specific linkage choice.

This approach is appropriate because the data is binary paint usage, and the palette is naturally interpreted as a set of used paints. Jaccard fits this set-overlap goal, so I treat Jaccard with hierarchical clustering as the primary clustering framework and then test sensitivity to linkage and dendrogram cutting choices.

In M4, I reran hierarchical clustering under multiple linkage methods while keeping the same basic representation. The cluster-size distributions shift a lot across methods (Table 1):

- Average linkage (Jaccard): 385 / 10 / 5 / 2 / 1
- Complete linkage (Jaccard): 331 / 44 / 15 / 13 (effectively 4 clusters under the cut)
- Single linkage: 399 / 1 / 1 / 1 / 1
- Ward: 155 / 74 / 66 / 58 / 50 (much more balanced)

I treat Ward and single-linkage mainly as "comparison points," because their behavior is driven by assumptions that don't match the set-overlap goal. Ward linkage is variance-based in Euclidean space; single linkage is prone to chaining.

What worked is that the basic story does not vanish once I move away from one exact setting. Under both average and complete (the two set-consistent linkages I care about most), there is still a clearly dominant cluster. What changes is the fine detail: the smaller clusters shift in size, and the exact membership of rare cases can move around. In other words, the big picture that there is a standard palette that covers most paintings is stable, but the exact composition of the rare clusters is linkage sensitive. That is why I keep average-linkage with Jaccard as the primary view, and I treat sensitivity as a result to report rather than something to hide. Single linkage is not useful here because it collapses into one chain-like cluster, and Ward is not directly comparable under the set-based interpretation.

3.2 Interpreting deviations: paint-combination structure via association rules

Clustering alone does not answer the most practical question raised in feedback: what paint combinations actually characterize the deviations? In this section, I use association rules to translate "clusters" into interpretable co-usage patterns, and I document how parameter choices affect rule volume.

The dataset matches the basket-setting exactly: each painting is a set of paints. Apriori-based rules provide directly interpretable co-occurrence statements (support/confidence/lift). However, the method is sensitive to thresholds, so parameter reporting is part of the analysis.

I mine global association rules after removing near-trivial paints. In particular, Titanium_White has global usage ≈ 0.992 (≥0.95), so keeping it tends to dominate rules without adding much interpretive value; removing it improves interpretability. I then vary min_support across 0.08, 0.10, and 0.15 (with other settings held fixed) and record the number of rules returned: 889 rules (0.08), 834 rules (0.10), and 734 rules (0.15). These results are summarized in the Appendix along with representative rule outputs.

Increasing min_support predictably reduces rule counts, which confirms that Apriori is threshold-sensitive. The main takeaway is not that one support value is "correct," but that I can transparently show how the candidate rule pool changes under stricter filtering. This also motivates using stricter thresholds for interpretation (so the report focuses on clearer co-usage signals) while keeping looser settings for sensitivity diagnostics. I do not treat rule volume changes as evidence that the underlying palette structure changes; rather, it reflects how aggressively the mining step filters patterns. Beyond rule volume, the within-cluster rules provide concrete signatures for the only non-dominant cluster with sufficient n (cluster 3, n=10). For example, rules involving Liquid_Clear with Black_Gesso (and related co-usage) appear with support = 0.30 (3/10 paintings), confidence = 1.00, and lift ≈ 2.0–2.5, which is meaningfully above independence. In contrast, the dominant cluster's rules largely mirror global co-usage and do not form distinctive cluster-only signatures under stricter thresholds.

3.3 External checks

I would like to check whether clusters align with season/episode or title text structure or not. If rare palette deviations correspond to specific seasons/episodes or specific title themes, that would support them as repeatable archetypes rather than arbitrary fragments. This time, I test alignment between palette clusters and external signals: metadata and title text.

Season/episode labels are direct external metadata. Titles are unstructured text, so a simple vectorization and clustering view offers an independent lens, even if titles turn out to be weak signals.

For metadata, I compute cluster x season (Table 2) and cluster x episode crosstabs (Table 4). For title themes (Table 3), I assign each painting a coarse title_theme label (landscape, water, season_time, objects_view, other) using keyword groups and then crosstab palette clusters against these themes. For a title-structure view, I vectorize titles using CountVectorizer (max_features=50, stop_words="english"), cluster the title vectors using KMeans (n_clusters=4, random_state=42), and crosstab title_cluster with palette cluster_labels.

The dominant palette cluster spans the full range of seasons/episodes, and also spans all title themes and title clusters, which is consistent with a general-purpose baseline palette rather than a period- or theme-specific mode. For the small clusters, sample sizes are too small to support strong claims about concentration, so I treat these checks as descriptive rather than decisive. Overall, these external checks support the interpretation that rare palette deviations are not simply "one

season" or "one title theme," but are more consistent with occasional palette variations within a highly consistent core template.

4. Findings & Interpretations

New insights from M4:

- The dominant palette is robust to cut and linkage choice.
- It does not align with a single season, episode, or title theme.
- Small clusters show distinct co-usage rules and can be interpreted as rare archetypes.
- Palette structure and title-based structure do not align one-to-one.
- Clustering alone gives candidate archetypes; rules, metadata, and title themes are needed to validate and interpret them.

Key insight 1: Bob Ross has a "standard palette" that dominates the series

Most episodes use basically the same core set of paints. So the series does not break into many equally common palette styles. It looks more like one default template that covers most paintings, with a small number of exceptions. For organizing a content library, this makes the "default" easy: one standard-palette category can cover the mainstream experience (good for general browsing and beginner pathways). It also means palette style alone won't create lots of large, distinct categories.

Key insight 2: The meaningful differences show up as rare "signature deviations," not as many balanced styles

Outside the baseline, variation shows up only occasionally. Many of the rare groups are tiny, so I don't think it's responsible to treat them as stable "types." Still, there is at least one rare group with enough episodes to support a consistent signature: when the palette changes, it tends to change in a specific way (a recognizable combination), not as random noise. This is useful for discovery. Keep the mainstream simple and surface a small number of "special palette" tags only where the evidence is strong.

Key insight 3: Rare palettes do not cleanly map to seasons or broad title themes

The baseline palette shows up throughout the run of the show, not in one particular season. The rare palettes also don't line up cleanly with broad title themes. Titles are just not a strong organizing signal here. If someone builds navigation around season or title keywords, they won't recover palette structure. Palette grouping adds a separate "style layer" that season/theme browsing doesn't capture.

However, there are some limitations. Several rare groups are extremely small, so they should be treated as candidates, not confirmed archetypes. A sensible next step is to test whether the same signature palette patterns repeat under alternative grouping choices and to connect palette changes to richer content descriptors than short titles.

Appendix

Shared GitHub Repository

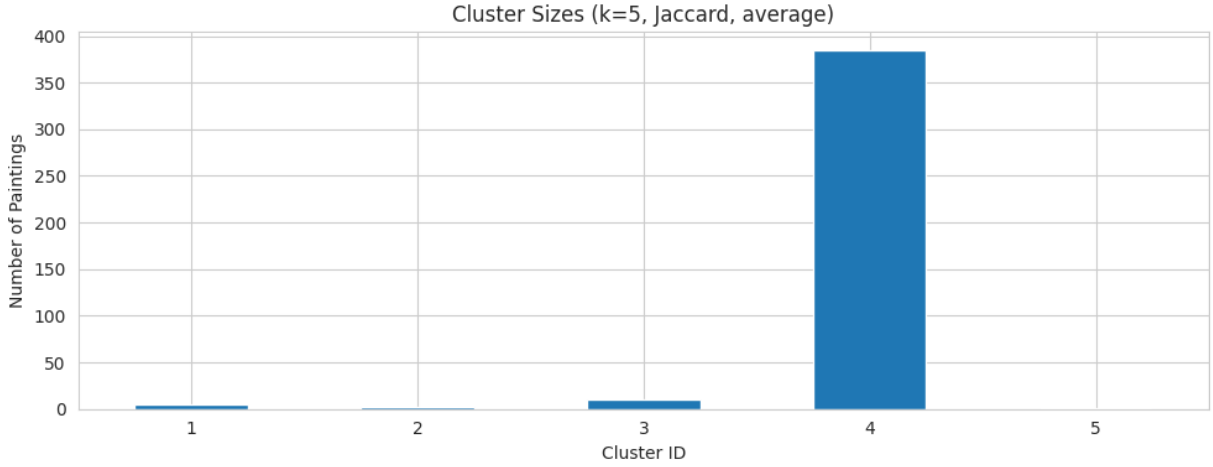
Link to shared team GitHub repository: https://github.com/yujiunzou/BA820-Unsupervised-ML-Project

My individual M4 work (where to find my files):

- Branch: "*Main*"
- Folder: "m4_individual_refinement"
- Folder: "*M4_Irene*"
- Primary notebook: "*M4_Irene/Irene_Tang_M4_Notebook.ipynb*"
- Supporting file (report PDF): "*M4_Irene/Irene_Tang_M4_Report.pdf*"

Notebook Link: https://github.com/yujiunzou/BA820-Unsupervised-ML-Project/blob/Irene-(Yihui)-Tang/M4_Irene/Irene_Tang_M4_Notebook.ipynb

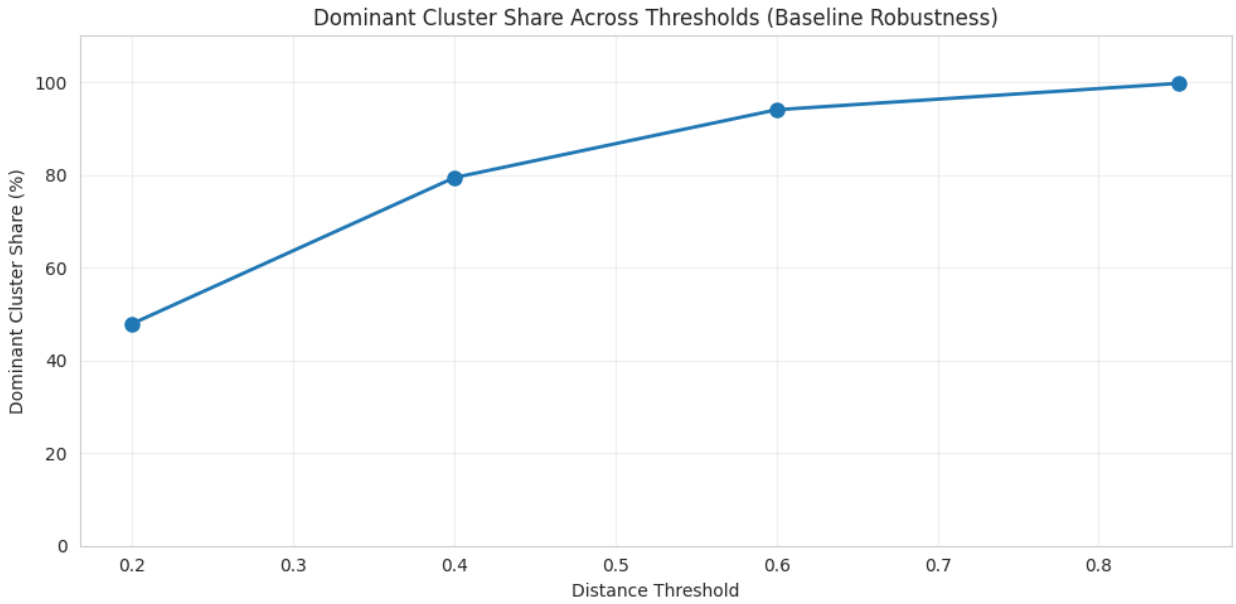
Supplemental Material



Plot 1: Cluster Sizes

	average	complete	ward	single
1	5	331.0	155	399
2	2	15.0	50	1
3	10	44.0	66	1
4	385	13.0	58	1
5	1	NaN	74	1

Table 1: linkage comparison size table



Plot 2: dominant share vs threshold

Season & Cluster Table:

season	1	2	3	4	5	6	7	8	9	10	...	22	23	24	25	26	27	28	29	30	31
cluster_labels																					
1	3	1	0	0	1	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
2	0	0	0	1	1	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	1	0	0	0	...	0	0	3	0	2	0	2	1	0	0
4	10	12	13	12	11	13	12	13	13	13	...	13	13	10	13	11	13	11	12	13	13
5	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0

5 rows × 31 columns

Table 2: Season & Cluster Table

Title theme & Cluster Table:

title_theme	landscape	objects_view	other	season_time	water
cluster_labels					
1	0	0	3	2	0
2	0	0	1	1	0
3	0	1	4	4	1
4	79	30	180	62	34
5	0	0	1	0	0

Table 3: Title theme & Cluster Table

Episode & Cluster Table:

episode	1	2	3	4	5	6	7	8	9	10	11	12	13
cluster_labels													
1	0	0	0	2	0	2	0	0	0	0	0	1	0
2	0	0	0	1	0	0	0	0	0	0	0	1	0
3	0	0	1	1	0	0	1	0	1	2	2	0	2
4	31	31	30	27	31	28	30	31	30	29	29	29	29
5	0	0	0	0	0	1	0	0	0	0	0	0	0

Table 4: Episode & Cluster Table

Process Overview

1. **Data loading**: Load the Bob Ross dataset (403 paintings) and select the 18 paint indicator columns.
2. **Data validation**: Confirm paint indicators are binary, with no missing values/duplicates in the feature matrix.
3. **EDA**: Summarize global paint usage rates and assess palette similarity using pairwise Jaccard similarity to confirm recurring palette templates.
   a) Variable Exploration
   b) Paint Columns
   c) Text Analysis
   d) Similarity
   e) EDA Summary
4. **Integrated Analysis**
   a) **Method 1: Hierarchical Clustering**
      i. Palette Clustering: Baseline and Robustness (Jaccard distance, agglomerative clustering, cut; linkage comparison for robustness)
      ii. Dominant share vs threshold
      iii. External Validation
         1. Cluster vs season, episode and title themes
         2. Linkage comparison
   b) **Method 2: Association rules (global & within-cluster):**
      i. Global Baseline Rules
      ii. Rule parameter sensitivity
      iii. With-cluster Rules
      iv. Interpretable Palette Rule
         1. Title-based clustering and comparison with palette cluster
5. **Insight generation**: Translate outputs into a non-technical interpretation: a dominant "Standard palette" archetype plus a small number of rare, structured archetypes with clear signature paint combinations.

Use of Generative AI Tools

Link: https://chatgpt.com/share/699cca63-5be8-8010-90db-926b485cd4ff

Summary:

I used ChatGPT as a support tool throughout M4, mainly for requirement organization, implementation guidance, method clarification, and final editing. It helped me turn the assignment instructions into a clearer checklist so I could track whether my M4 refinements (robustness checks, external checks, and interpretation updates) were aligned with the requirements.

For implementation questions, I asked how to examine relationships between palette clusters and season / episode / title-theme labels in Python, and ChatGPT suggested using pd.crosstab() with example code structure. I then wrote and ran the crosstab analyses myself and interpreted the outputs in the notebook.

I also used ChatGPT to understand one output issue in the linkage-comparison table: under complete linkage, I requested 5 clusters, but one column showed NaN. ChatGPT explained that under some dendrogram structures, the cut can produce fewer effective non-empty clusters than the requested maximum, so the NaN reflected the clustering result format (table alignment) rather than a coding bug. This helped me report the result correctly.

For method decisions, I discussed whether I should keep average linkage or switch to complete linkage after seeing that both produced one dominant cluster plus several small clusters. I also asked how to interpret linkage sensitivity across average, complete, single, and Ward linkage. ChatGPT prompted me to think about the data representation (binary paint usage as set overlap), which helped me retain average linkage with Jaccard as the primary view while reporting sensitivity as part of the findings. It also helped me frame Ward as a comparison point rather than the primary basis for interpretation.

For the association-rule sensitivity section, I asked how to set min_support thresholds in a more professional way. ChatGPT suggested using multiple thresholds as a sensitivity analysis, and I implemented that comparison myself.

I also asked implementation-level questions about how to summarize and visualize a 403×403 cosine similarity matrix more clearly in Python (e.g., upper-triangular summaries, filtering non-zero values, and summary statistics instead of plotting the full dense matrix directly). Based on that discussion, I implemented the code and interpreted the results myself.

Finally, I used ChatGPT for grammar checks and a final pass to confirm that my write-up covered the M4 submission requirements. ChatGPT did not run my notebook or generate my final results. All coding, analysis execution, output validation, and final interpretations were completed by me.

Coding Structure from ChatGPT that I used:

```python
ct_counts = pd.crosstab(df[SEASON_COL], df[CLUSTER_COL])  # rows: season, cols: cluster
ct_rowprop = ct_counts.div(ct_counts.sum(axis=1), axis=0) # P(cluster | season)
ct_colprop = ct_counts.div(ct_counts.sum(axis=0), axis=1) # P(season | cluster)

display(ct_counts.head())
display(ct_rowprop.head())

# Optional: save for report appendix
# ct_counts.to_csv("cluster_by_season_counts.csv")
# ct_rowprop.to_csv("cluster_by_season_rowprop.csv")
```

Python

```python
# convert upper triangle to a long table (avoid duplicates and diagonal)
sim = cos_sim_df.copy()

# use original title labels (or keep numeric ids + title column separately
arr = sim.values
n = arr.shape[0]

pairs = []
for i in range(n):
    for j in range(i+1, n):  # upper triangle only
        pairs.append((sim.index[i], sim.columns[j], arr[i, j]))

pairs_df = pd.DataFrame(pairs, columns=["title_i", "title_j", "cosine_sim"
pairs_df = pairs_df.sort_values("cosine_sim", ascending=False)

# show top 20 non-identical title pairs
pairs_df.head(20)
```

Python

```python
upper_nz = upper[upper > 0]
pd.Series(upper_nz).describe()
```