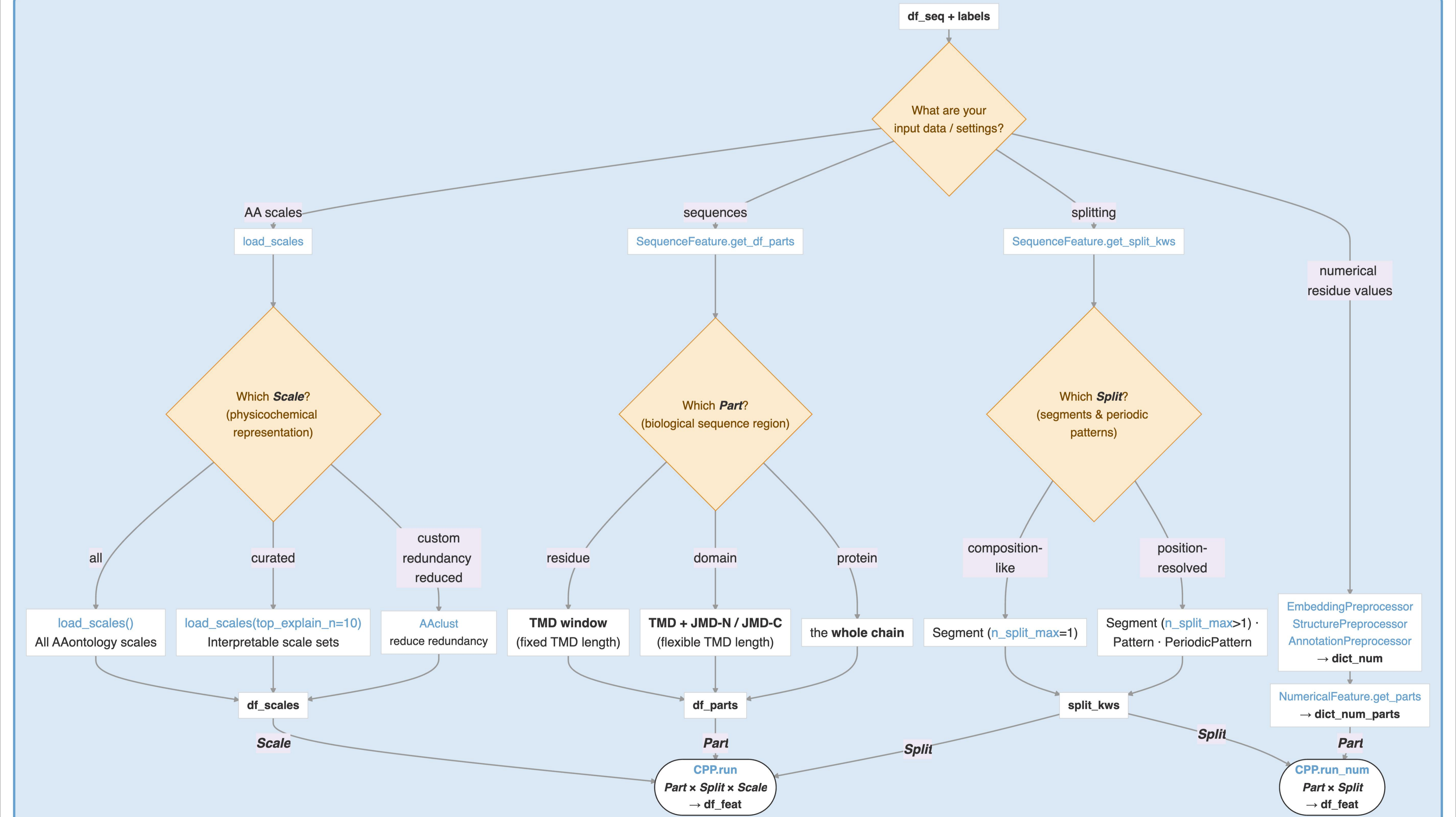


CPP Feature engineering (Parts × Splits × Scales): How AAanalysis identifies interpretable physicochemical features



CPP needs two inputs: features (Parts × Splits × Scales, built in the CPP block below) and labels (test group = 1 vs reference group = 0). Sampling builds the label-0 reference group: use your own negatives, AAWindowSampler decoys (residue level), or dPULearn — a deterministic PU-learning (ML) step that mines reliable negatives from positives + an unlabeled pool by PCA / distance on a feature matrix X (embeddings or features). It produces labels and is independent of CPP. Splits apply to both df_parts and dict_num parts — only Scales are sequence-only; numerical inputs already carry their values, so they skip Scales and go through CPP.run_num. Two axes of a prediction task: kind of task (classification / regression / multi-class) × kind of data / unit (residue-window / domain-TMD / whole protein). Explain: CPPPlot visualizes the CPP group signature straight from df_feat (no model); TreeModel (global importance) and ShapModel (local SHAP) need a fitted model on X. Colors: blue = class / method / function (cheat-sheet style); dashed = dev-preview / planned (post v1.1). Plot pairs: every analysis class has a mirror *Plot.