FSLSM Learning Style AI - Architecture Description

Section 1 : Project Overview

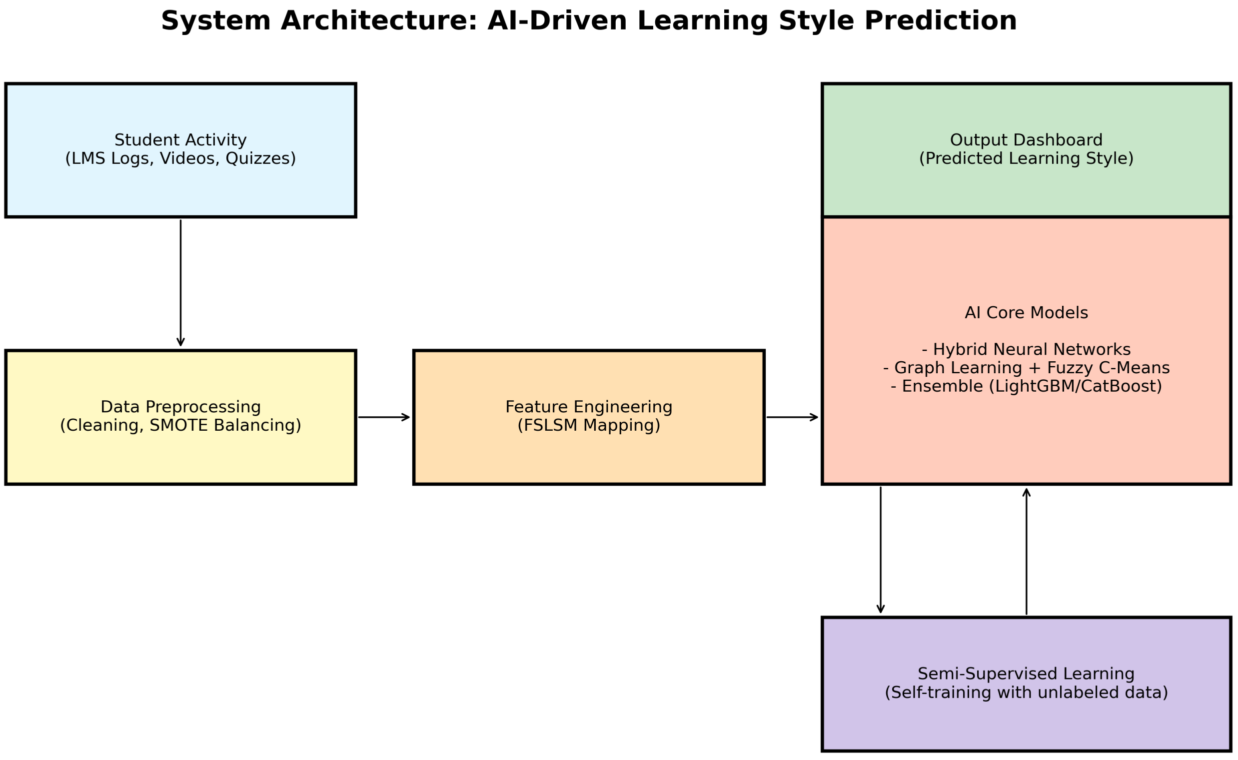
The Learning Style AI is a sophisticated educational technology application designed to identify student learning styles based on the Felder-Silverman Learning Style Model (FSLSM). Unlike traditional rule-based questionnaires, this system employs Semi-Supervised Learning (SSL) and State-of-the-Art (SOTA) Deep Learning models to predict learning styles from behavioral data.

The system serves two primary user roles:
- Students: Receive personalized learning style analysis and study recommendations.
- Faculty: Perform batch analysis of class data to understand learning trends.

Section 2 : High-Level Architecture

The system follows a modern Model-View-Controller (MVC) inspired pattern, adapted for a Data Science application.

It integrates a Streamlit Frontend for UI, a sophisticated Inference Engine for backend logic, and a Data Layer managing CSVs and Model serialization.



Section 3 : Machine Learning Architecture (The Core)

The system's distinguishing feature is its Semi-Supervised Competitive Training Pipeline. It does not rely on a single algorithm; instead, it trains multiple advanced models and dynamically selects the best performer for each of the four FSLSM dimensions.

3.1. The Training Pipeline
This pipeline handles limited labeled data by leveraging unlabeled data via Self-Training.
- Optimization: Optuna is used to find optimal hyperparameters for XGBoost.
- Self-Training: XGBoost and CatBoost are wrapped in a Self-Training Classifier. They iteratively label high-confidence data points in the unlabeled set and retrain themselves.
- Pseudo-Labeling: TabNet (a deep learning model for tabular data) is trained using the pseudo-labels generated by the best gradient boosting model.

3.2. Model Zoo
The architecture incorporates cutting-edge algorithms:
- KAN (Kolmogorov-Arnold Network): Uses B-Splines on edges to capture complex, non-linear relationships.
- TabNet: Uses Sequential Attention to mimic decision trees within a Neural Network.
- NAM (Neural Additive Model): Acts as a "Glass Box" model for transparent decision making.
- XGBoost/CatBoost: Robust baselines and "Teacher" models for TabNet.

3.3. Feature Engineering
- Input Mapping: 17 raw behavioral inputs are mapped to the 4 FSLSM dimensions.
- Fuzzy C-Means (FCM): Adds "Cluster Membership" features to help distinguish non-linear groupings.
- SMOTE: Used to handle class imbalances.

Section 4 : Frontend & Application Logic

4.1. Streamlit Dashboard
- Multi-Role: Renders Student or Faculty views based on authentication state.
- Real-time Inference: Constructs feature vectors from user sliders and loads specific SOTA models for each dimension.
- Fallback Logic: Ensures app stability using rule-based calculations if models fail to load.
- Visualization: Renders interactive Radar Charts and generating professional PDF reports.

Section 5 : Data Flow

1. Input: User interaction (Sliders) or Batch Upload (CSV).
2. Vectorization: Data is normalized and mapped to the training feature schema.
3. Prediction: The FSLSMPredictor iterates through all 4 dimensions, calling predict_proba() on the loaded SOTA model.
4. Confidence Calculation: Computes certainty metrics based on distance from the decision boundary.
5. Output: Displayed as UI Cards, Graphs, and generated PDF.

Section 6 : Deployment & Tech Stack

- Language: Python 3.9+
- Framework: Streamlit
- ML Libraries: PyTorch (KAN, TabNet), Scikit-learn, CatBoost/XGBoost, Imbalanced-learn.
- Containerization: Docker standard Python image.