	Script Name	Purpose	Inputs	Outputs
DATA PREPARATION	**000_config.R**	Centralized configuration for the H1 R data prep pipeline. Defines global paths, libraries, and simulation constraints (Constraint-Aware Replication).	-	-
	001_data_import_cleaner.R	Clean and synchronize raw origination and performance .txt files for analysis. Steps include reading raw files, removing NA delinquency, applying credit filters (FICO/LTV/DTI, Default Timer), and standardizing formats.	• historical_data_YYYYQ# • historical_data_time_YYYYQ#.txt • 000_config.R	• orig_data_cleaned.rds • perf_data_cleaned.rds
	002_data_prep_feature.R	Prepare model-ready features from cleaned data. Includes sampling (250k loans, 5% default), stratification engineering (geo/lender), leakage-safe capping/imputation, and encoding binary indicators.	• orig_data_cleaned.rds • perf_data_cleaned.rds • 000_config.R	• final_features_h1.rds
	003_prep_split_scale.R	Define strict train/test time windows, derive prediction targets (12-month horizon), mask invalid loans, and compute leakage-safe scaling parameters based on training data.	• final_features_h1.rds • 000_config.R	• train_targets.rds • test_targets.rds • preproc_params.rds • final_features_h1.rds
	004_validation_checks.R	Comprehensive validation suite for H1 data pipeline outputs. Checks file existence, dimensions, split integrity, target distribution, feature correctness, and graph construction readiness.	• orig_data_cleaned.rds • perf_data_cleaned.rds • final_features_h1.rds • train_targets.rd • test_targets.rds • preproc_params.rds	• validation_summary_h1.csv
	005_EDA.Rmd	Exploratory Data Analysis (EDA) report. Validates methodological constraints (raw features, 5% default rate) and assesses predictive signal quality before modeling.	• 000_config.R • final_features_h1.rds • train_targets.rds	• 005_EDA.html

	Script Name	Purpose	Inputs	Outputs
MODEL PIPELINE	**config_h1.py**	Centralized configuration file defining paths, hyperparameters, temporal windows, and topology settings used across all H1 scripts.	-	-
	python_data_loader_h1.py	Implements the core H1 data pipeline (Static Supra-Graph & rolling windows). Includes	• final_features_h1.rds	• train_features.pt

		burn-in recovery for full training history and hybrid connectivity with stochastic sampling for large groups.	• train_targets.rds • test_targets.rds • preproc_params.rds	• static_edge_index.pt
	baseline_data_prep_h1.py	Prepare constraint-aligned datasets for H1 baseline models by enforcing identical temporal splits and topology definitions used in the dynamic H1 model.	• train_features.pt • test_features.pt • final_features_h1.rds	• baseline_data_h1_aligned.pkl
	model_architecture_h1.py	Defines the DYMGNN architecture (GAT → LSTM → Attention → Decoder). Includes stability mods and replaces Zandi et al.'s (2025) problematic Eq. 7 pooling with a Bahdanau-style MLP.	• train_features.pt • test_features.pt • static_edge_index.pt • metadata.pt	• best_model_h1.pt
	training_h1.py	Orchestrates DYMGNN training using a 'Structure-First' strategy. Implements Thesis Sec 3.2.3 protocols including Pos Loss, Mixed Precision, and Ghost Node filtering.	• train_features.pt • static_edge_index.pt • metadata.pt	• best_model_h1.pt • training_results.jso • TensorBoard Logs
	baseline_models_h1.py	Train and evaluate all H1 baseline models (non-GNN and static GNN) to establish a performance benchmark under the same topology and temporal constraints.	• baseline_data_h1_aligned.pkl	• baseline_results_h1.json
	evaluation_h1.py	Perform final empirical evaluation of the H1 model by generating predictions, computing the full metric suite, and benchmarking against static baselines.	• best_model_h1.pt • test_features.pt • static_edge_index.pt • baseline_predictions.pkl	• h1_predictions.pkl • calibration_curve_comparison.png • Console report

	Script Name	**Purpose**	• **Inputs**	• **Outputs**
VALIDATION & ANALYSIS	**topology_analysis.py**	Analyze stochastic block graph topology to validate design choices. Evaluates group sizes/degree distributions to recommend NeighborLoader sampling parameters and estimate VRAM requirements.	• final_features_h1.rds • config_h1.py	• Console Output (Diagnostics, Degree Stats, VRAM Estimates)
	diagnostics_h1.py	Validate the H1 data pipeline, graph structure, and core model mechanics through targeted diagnostic checks.	• train_features.pt • static_edge_index.pt • metadata.pt	• Console diagnostics

			• train_targets.rds	
	attention_weights.py	Extract and visualize temporal attention weights from a trained DYMGNN model to analyze global and node-level attention patterns.	• test_features.pt • static_edge_index.pt • best_model_h1.pt	• attention_plot.png • attention_samples.png
	interpretability_h1.py	Provide post-hoc interpretability for the trained H1 DYMGNN model using SHAP (KernelExplainer).	• test_features.pt • best_model_h1.pt • metadata.pt • config_h1.py	• h1_shap_summary.png • h1_shap_dep_<feature>.png

	Script Name	Purpose	Inputs	Outputs
DATA PREPARATION	**00_config.R**	Centralized configuration for the H2 R data prep pipeline. Defines global paths, libraries, and specific parameters for the Proposed Model (H2), ensuring separation from H1 constraints.	–	–
	01_data_import_cleaner.R	Import, clean, and synchronize raw Freddie Mac origination and performance data for the H2 pipeline. Steps include reading raw files, removing NA delinquency, applying credit filters (FICO/LTV/DTI, Default Timer), and syncing.	• Raw Freddie Mac .txt files • 00_config.R	• orig_data_cleaned.rds • perf_data_cleaned.rds
	02_data_prep_feature.R	Construct balanced modeling dataset (~250k loans, 5% default). Steps include stratified sampling, outlier capping, missing value imputation, and engineering default timing/loan-age features.	• orig_data_cleaned.rds • perf_data_cleaned.rds	• final_features_raw_clean.rds
	03_prep_split_scale.R	Prepare final feature set for modeling. Engineers UPB and modification features, derives targets, splits train/test sets, and computes scaling parameters. Ensures consistency with H1 pipeline artifacts.	• final_features_raw_clean.rds	• preproc_params.rds • train_targets.rds • test_targets.rds • final_data_base_with_targets.rds
	04_validation_checks.R	Comprehensive validation suite for H2 pipeline. Checks file existence, time window integrity (Jan 2012–Jun 2013), target distributions, H2 engineered features (e.g., UPB %), and graph readiness.	• 00_config.R • orig_data_cleaned.rds • perf_data_cleaned.rds • final_features_h2.rds • train_targets.rds • test_targets.rds • preproc_params.rds	• validation_summary.csv • target_check.csv
	05_EDA.Rmd	Exploratory Data Analysis (EDA) report for H2. Validates H2-specific features (e.g., UPB ratios), assesses credit driver signals (FICO/LTV), analyzes Interest Rate Paradox, and verifies graph topology readiness.	• 00_config.R • final_features_raw_clean.rds • final_data_base_with_targets.rds • train_targets.rds	• 05_EDA.html

	Script Name	Purpose	• Inputs	• Outputs
VALIDATION & ANALYSIS	**diagnostics.py**	Strict diagnostic suite for H2. Validates R/Python alignment, normalization, and temporal splits. Checks dynamic graph consistency/density and verifies model mechanics via synthetic forward passes and tiny batch overfitting tests.	• train_graphs.pt • test_graphs.pt • train_targets.rds • config_2.py	• Console diagnostics
	interpretability_h2.py	Provide post-hoc interpretability for H2 GNN using SHAP. Wraps DT-GNN for KernelExplainer (including synthetic graph construction), flattens temporal features, uses K-Means background, and aggregates SHAP values.	• test_graphs.pt • best_model_final.pt • config_2.py	• h2_shap_summary.png • h2_shap_dep_<feature>.png

	Script Name	Purpose	Inputs	Outputs
MODEL PIPELINE	**config_2.py**	Centralized configuration file for H2. Defines paths, hyperparameters, temporal windows, and topology settings specific to the H2 "Implicit" model and baselines.	–	–
	python_data_loader_h2.py	Implements data engineering for H2. Constructs DYNAMIC graphs where connectivity is defined by behavioral similarity (k-NN) rather than static metadata. Re-computes topology at every snapshot.	• final_data_base_with_targets.rds • config_h2.py	• train_graphs.pt • test_graphs.pt
	baseline_data_prep_h2.py	Prepares aligned datasets for H2 Baselines (XGBoost, LR, Static GNNs) using "Implicit" feature set and "Behavioral" topology. Enforces H1 temporal splits and uses temporal averaging to collapse sequences.	• train_graphs.pt • test_graphs.pt • final_data_base_with_targets.rds • config_h2.py	• train_graphs.pt • test_graphs.pt • graph_metadata.pkl
	model_architecture_h2.py	Defines H2 Dynamic Temporal GNN (Implicit Topology). Features TemporalGAT (with residuals), Unidirectional LSTM, and Supra-Graph Pooling (Geo+Lender) per Thesis Sec 3.3.	• snapshot_sequence (PyG Objects) • config_h2.py	• logits (Tensor in memory)
	training_h2.py	Trains the H2 DT-GNN using implicit behavior-driven topology. Implements Focal Loss for class imbalance, dynamic class weighting, and validation via AUC/F1. Saves best checkpoint.	• train_graphs.pt • config_h2.py	• best_model_final.pt • training_results.json • TensorBoard Logs
	baseline_models_h2.py	Trains/evaluates static baselines for H2 using "Implicit" KNN-based graphs. Tests behavior > metadata hypothesis. Replicates Thesis logic: dynamic KNN construction, 50% random node isolation during training, and specific 4-layer DNN architecture.	• baseline_data_h2_aligned.pkl	• baseline_results_h2.json
	evaluation_h2.py	Conducts final assessment of H2 Implicit Model. Computes metrics (AUC, F1 @ 0.5, Brier) and benchmarks against baselines. Performs statistical tests (DeLong, McNemar) per Thesis Sec 4.1 & 4.2.	• best_model_final.pt • test_graphs.pt • baseline_predictions_h2.pkl	• h2_predictions.pkl • calibration_curve_comparison.png • Console Report