Credit Delinquency Prediction — Model Plan for Geldium

Dataset analysed: Delinquency_prediction_dataset.csv
Report generated: 2025-08-08 12:18:01
Rows: 500, Columns: 19
Detected target column: Delinquent_Account
Target positive rate (1): 0.16
Target counts: {0: 420, 1: 80}

Exploratory Data Analysis (summary)
- Top 5 columns by data type:
  1. Customer_ID (object)
  2. Age (int64)
  3. Income (float64)
  4. Credit_Score (float64)
  5. Credit_Utilization (float64)
  6. Missed_Payments (int64)
  7. Delinquent_Account (int64)
  8. Loan_Balance (float64)
  9. Debt_to_Income_Ratio (float64)
  10. Employment_Status (object)
  11. Account_Tenure (int64)
  12. Credit_Card_Type (object)

- Missing value summary (top columns):
  Income: 39 missing (7.8%)
  Loan_Balance: 29 missing (5.8%)
  Credit_Score: 2 missing (0.4%)

- Method used to pick top features: numeric correlation with the target (absolute correlation) where available, otherwis
e numeric variance.

Modeling options and pipeline

1) Two suggested modeling options:
   - Simple baseline: Logistic Regression (with L2 regularization) — interpretable, fast, and regulatory-friendly.
   - Complex/Production: Gradient-Boosted Trees (LightGBM / XGBoost / CatBoost) — higher predictive power, handles no
inearities and interactions.

2) General predictive pipeline (from data ingestion to deployment):
   - Data ingestion: read from source, validate schema, apply basic type coercion.
   - Data cleaning: impute missing values (median for numeric, mode or dedicated 'missing' category for categorical), id
entify and cap outliers.
   - Feature engineering: create credit utilization ratios, rolling missed-payment counts, time-since-last-default, inco
me-to-debt ratios, categorical grouping (occupation, region).
   - Encoding: one-hot or target encoding for high-cardinality categorical variables; scale numeric features for linear
models.
   - Train/validation split with temporal holdout if possible (e.g., train on older loans, validate on newest) to avoid
lookahead bias.
   - Model training: baseline logistic regression and a GBT ensemble.
   - Model calibration: isotonic or Platt scaling to ensure well-calibrated probabilities.
   - Explainability: SHAP values (for tree models) and coefficient tables (for logistic regression) for regulatory trans
parency.
   - Monitoring & retraining: drift detection, periodic retraining, and performance/fairness dashboards.

3) How customer inputs transform into a final risk score (high level):

- Ingest a customer's record -> apply same preprocessing and feature engineering -> pass features to the trained model -> model outputs a probability of delinquency -> probability is calibrated and mapped to risk bands (low/medium/high) -> generate explanation (top contributors) and recommended action.

4) Pseudocode for pipeline
```
# Pseudocode
raw = read_source()
clean = impute_missing(raw)
features = engineer_features(clean)
X_train, X_val, y_train, y_val = temporal_split(features)
# Baseline
lr = LogisticRegression()
lr.fit(X_train, y_train)
# Production model
gbt = LGBMClassifier()
gbt.fit(X_train, y_train)
# Evaluate, calibrate, and store model and explainer
shap_explainer = shap.TreeExplainer(gbt)
# Deploy: save preprocessing pipeline + model + explainer
```

Chosen model (summary) — 2–3 sentences
I recommend a Gradient-Boosted Tree (LightGBM/XGBoost) as the primary production model, with a logistic regression baseline for interpretability and regulatory comparisons. GBTs balance strong predictive power with well-developed explainability tools (SHAP), and handle missing values, non-linearities and feature interactions common in credit data.

Top 5 input features considered most informative (based on available correlations/variance in this dataset):
1. Income
2. Account_Tenure
3. Credit_Score
4. Debt_to_Income_Ratio
5. Credit_Utilization

Justification for model choice:
Gradient-Boosted Trees deliver superior predictive performance in credit risk tasks because they capture non-linear patterns and feature interactions (e.g., income vs utilization vs missed payments). Using a GBT alongside a logistic-regression baseline provides a practical balance: maintain interpretability for regulatory reporting and use SHAP or surrogate models to explain GBT outputs. This approach meets Geldium's likely needs for accuracy, explainability, and operational scalability while supporting compliance by producing human-readable explanations and calibration.

Evaluation strategy

Key metrics:
- Discrimination: ROC-AUC (overall), PR-AUC (esp. for imbalanced positive class), Gini coefficient
- Classification metrics: Precision, Recall, F1 at business-operational thresholds (e.g., top X% highest risk), Confusion Matrix
- Calibration: Brier score, calibration plots, and calibration-in-the-large; use isotonic/Platt if needed
- Business metrics: population/stability-weighted lift, PD deciles, expected loss estimates

Fairness and bias checks:
- Monitor metrics (AUC, precision, recall, calibration) across protected groups (gender, age bands, geography, income bands).
- Check for disparate impact (selection rate ratios), equalized odds differences, and calibration parity by subgroup.
- If bias detected: consider pre-processing (reweighing), in-processing (fairness-constrained optimization), or post-processing (threshold adjustments), and document decisions.

Interpretation & action thresholds:
- Define risk bands (e.g., Low < 5% PD, Medium 5–20%, High >20%) based on calibrated PD and business appetite.
- Flag model for retraining if AUC drops > 3–5 points vs baseline, or if calibration slope shifts significantly, or if subgroup performance deteriorates beyond defined SLAs.

Monitoring & lifecycle:
- Set up automated monitoring for model performance, data drift, and feature distribution shifts; log predictions and outcomes for backtesting; schedule monthly or quarterly reviews and ad-hoc retraining when triggered.

Next steps & recommendations

- Create a reproducible training pipeline with versioned preprocessing, models, and hyperparameters.
- Implement SHAP-based explainability and produce regulator-friendly explanation templates.
- Build monitoring dashboards for performance and fairness metrics, and define retraining governance.
- Validate with a temporal split (or nested CV) and run stress tests for economic downturn scenarios.

Appendix: first 5 rows preview

Customer_ID	Age	Income	Credit_Score	Credit_Utilization	Missed_Payments	Delinquent_Account	Loan_Balance	Debt_to_Income_Ratio	Employment_Status	Account_Tenure	Credit_Card_Type	Location	Month_1	Month_2	Month_3	Month_4	Month_6
0 CUST0001	56	165580.0	398.0	0.390501928600237	3	0	16310.0	0.31739611496857	EMP	18	Student	Los Angeles	Late	Late	Missed	Late	Missed	Late
1 CUST0002	69	100999.0	493.0	0.312444033226017	6	1	17401.0	0.1960926 43360821	Self-employed	0	Standard	Phoenix	Missed	Missed	Late	Missed	On-time	On-time
2 CUST0003	46	188416.0	500.0	0.359930238508409	0	0	13761.0	0.3016549 39890911	Self-employed	1	Platinum	Chicago	Missed	Late	Late	On-time	Missed	Late
3 CUST0004	32	101672.0	413.0	0.371400355303824	3	0	88778.0	0.2647944 39228806	Unemployed	15	Platinum	Phoenix	Late	Missed	Late	Missed	Late	Late
4 CUST0005	60	38524.0	487.0	0.234715862628201	2	0	13316.0	0.5105834 82534848	Self-employed	11	Standard	Phoenix	Missed	On-time	Missed	Late	Late	Late

Column types:
Customer_ID: object
Age: int64
Income: float64
Credit_Score: float64
Credit_Utilization: float64
Missed_Payments: int64
Delinquent_Account: int64
Loan_Balance: float64
Debt_to_Income_Ratio: float64
Employment_Status: object
Account_Tenure: int64
Credit_Card_Type: object
Location: object
Month_1: object
Month_2: object
Month_3: object
Month_4: object

Month_5: object
Month_6: object