We used a **Random Forest Classifier**, a robust ensemble model that builds multiple decision trees and combines their outputs to improve accuracy and reduce overfitting. It is well-suited for structured data, handles nonlinear relationships effectively, and can identify key predictors even when some data is missing or imbalanced.

### ⭐ Top 5 Input Features (Based on Feature Importance):

1. **Missed\_Payments** – Directly reflects a customer’s past payment behavior, a strong indicator of future delinquency.
2. **Credit\_Score** – Captures overall creditworthiness; lower scores are typically linked to higher delinquency risk.
3. **Income** – Influences repayment capacity; lower or unstable income can raise the risk of missed payments.
4. **Credit\_Utilization** – High utilization suggests over-reliance on credit and potential financial strain.
5. **Loan\_Balance** – Higher outstanding balances may indicate debt stress and increase likelihood of default

#### ****Justifying model choice****

### Logistic Regression

#### Pros:

* **Simple & Interpretable**: Easy to explain to stakeholders and regulators—shows how each variable affects the odds of delinquency.
* **Fast to Train**: Computationally efficient, even with large datasets.
* **Well-Suited for Linearly Separable Data**: Works well when the relationship between features and the target is linear.
* **Handles Probabilities Naturally**: Directly outputs a probability of risk, useful in scoring systems.

#### Cons:

* **Limited to Linear Relationships**: Can’t model complex patterns without feature engineering.
* **Sensitive to Outliers**: Extreme values can bias results unless treated properly.
* **Assumes Independence**: Assumes features are not strongly correlated, which may not hold in financial datasets.

### 🔹 Decision Trees

#### Pros:

* **Captures Nonlinear Relationships**: Automatically models complex interactions between variables.
* **Intuitive Flow**: Easy to visualize (e.g., "if income < ₹20,000 and 2 missed payments → high risk").
* **Handles Missing Values Well**: Some implementations can split data even with nulls.
* **No Feature Scaling Needed**: Works with raw data without normalization.

#### Cons:

* **Prone to Overfitting**: Single decision trees can fit noise unless pruned or regularized.
* **Less Interpretable for Deep Trees**: Can become complex and harder to explain at scale.
* **Instability**: Small changes in data can result in very different trees.

### Summary:

* **Use Logistic Regression** when interpretability, regulatory compliance, and speed are top priorities.
* **Use Decision Trees** when your data is complex or non-linear and you want more flexible modeling.

| **Model Type** | **Performance** | **Explainability** | **Summary** |
| --- | --- | --- | --- |
| **Logistic Regression** | ⭐⭐ (Good for simple patterns) | ⭐⭐⭐⭐ (Highly interpretable) | Best when transparency and regulatory compliance are key. |
| **Decision Tree** | ⭐⭐⭐ (Handles non-linear data) | ⭐⭐⭐ (Visualizable, easy rules) | Good tradeoff for small to medium datasets, interpretable with moderate performance. |
| **Random Forest** | ⭐⭐⭐⭐ (Strong, robust model) | ⭐⭐ (Feature importances only) | High accuracy; harder to explain full decision path but more stable than a single tree. |
| **XGBoost / Gradient Boosting** | ⭐⭐⭐⭐⭐ (State-of-the-art) | ⭐ (Complex to explain manually) | Excellent accuracy; requires SHAP or LIME for interpretation. |
| **Neural Networks (MLP)** | ⭐⭐⭐⭐ (Good with large data) | ⭐ (Black-box model) | High capacity, but very hard to interpret for financial justification. |

Here’s how common **credit delinquency prediction models** align with operational needs such as **speed**, **scalability**, and **ease of monitoring** in a real-world banking or lending environment:

## **1. Logistic Regression**

**Speed**: ⚡ Very fast for both training and prediction, even with large datasets.  
**Scalability**: ✅ Scales well to millions of rows; low computational cost.  
**Ease of Monitoring**: ⭐⭐⭐⭐⭐ Very easy—coefficients directly explain variable impact, making it ideal for ongoing compliance checks.  
**Fit**: Great for **real-time scoring** and environments needing **clear audit trails**.

## **2. Decision Trees**

**Speed**: 🚀 Fast training and prediction for small to medium datasets; may slow down with very large data.  
**Scalability**: ✅ Handles moderate scaling well; deep trees can become heavy.  
**Ease of Monitoring**: ⭐⭐⭐ Clear, visual rules; easy to explain decisions to auditors or risk officers.  
**Fit**: Good for **medium-scale batch scoring** where transparency is still important.

## **3. Random Forest**

**Speed**: ⚖️ Slower than single trees, especially for large ensembles.  
**Scalability**: ✅ Parallelizable, so can handle large datasets on distributed systems.  
**Ease of Monitoring**: ⭐⭐ Only feature importance available natively; needs extra tools for deeper explanation.  
**Fit**: Ideal for **large-scale batch scoring** where accuracy is more important than instant interpretability.

## **4. XGBoost / Gradient Boosting**

**Speed**: ⏳ Slower to train but can be optimized for fast prediction with tuned parameters.  
**Scalability**: ✅ Highly scalable with GPU or distributed computing support.  
**Ease of Monitoring**: ⭐ Needs tools like **SHAP** or **LIME** for explainability; less transparent out-of-the-box.  
**Fit**: Best for **high-volume risk scoring** where maximizing accuracy outweighs instant transparency.

## **5. Neural Networks (MLP)**

**Speed**: ⏳ Slowest to train; prediction speed depends on architecture.  
**Scalability**: ✅ Excellent with cloud/GPU infrastructure.  
**Ease of Monitoring**: ⭐ Very low—black-box nature makes it challenging for regulated credit decisions.  
**Fit**: Suitable for **non-regulated risk assessment** where extreme accuracy is required and interpretability is less critical.

#### ****Plan how to evaluate model performance****

| **Metric** | **Description** |
| --- | --- |
| **Accuracy** | Overall percentage of correct predictions. May be misleading in imbalanced data. |
| **Precision** | % of predicted high-risk customers who were actually high-risk. Important for reducing false alarms. |
| **Recall (Sensitivity)** | % of actual high-risk customers the model correctly identified. Critical for catching risky cases. |
| **F1-Score** | Harmonic mean of precision and recall—balances both. |
| **AUC-ROC** | Measures how well the model distinguishes between good and bad customers. Higher is better. |
| **Confusion Matrix** | Provides true/false positives and negatives—useful for visualizing model behavior. |

## **Performance Metrics**

### ✅ **Accuracy**

* **What it measures**: Proportion of total correct predictions.
* **Use case**: Basic model sanity check.
* **Interpretation**: Can be misleading with imbalanced classes; should not be relied on alone.

### 🎯 **F1 Score**

* **What it measures**: Harmonic mean of precision and recall.
* **Use case**: Balancing false positives and false negatives in delinquency detection.
* **Interpretation**: A high F1 score means the model correctly flags delinquent customers without over-penalizing safe ones.

### 📈 **AUC-ROC (Area Under Curve - Receiver Operating Characteristic)**

* **What it measures**: Ability to distinguish between delinquent and non-delinquent customers at various thresholds.
* **Use case**: Evaluate overall model discrimination power.
* **Interpretation**: Closer to 1.0 is best; >0.80 is generally strong.