### **1**Drop original features if the new ones outperform them

### **8. Outlier Detection**

#### **✅ Why:**

* Outliers can distort model learning and metrics.

#### **🔍 What to look for:**

* Extremely high balances or incomes
* Sudden jumps in utilization

#### **🛠️ Solutions:**

* Cap/floor values (e.g., 1st–99th percentile)
* Use robust models or robust scalers

### **9. Class Imbalance Handling**

#### **✅ Why:**

* Charge-off is likely rare (<5%), and models may ignore it.

#### **🔍 What to do:**

* Oversample minority class (e.g., SMOTE)
* Undersample majority class
* Use weighted loss functions

#### **🛠️ Solutions:**

* Try multiple strategies and compare performance on recall, F1, etc.
* Always validate on real (imbalanced) distribution

### **10. Data Leakage Checks**

#### **✅ Why:**

* Leakage leads to unrealistic performance in training that won’t hold in production.

#### **🔍 What to check:**

* Any column that could contain future info or derived from the target  
  **. Basic Data Quality Checks**

#### **✅ Why:**

* To understand the shape, data types, and overall structure before deeper analysis.

#### **🔍 What to look for:**

* Incorrect data types (e.g., float for binary)
* Unexpected number of columns/rows

#### **🛠️ Solutions:**

* Convert data types (e.g., .astype('int'))
* Rename or drop irrelevant columns

### **2. Missing Values**

#### **✅ Why:**

* Missing data can bias results or cause models to fail.

#### **🔍 What to look for:**

* Columns with many nulls (e.g., FICO might be null for “No-Hit” cases)
* Target columns with unexpected nulls

#### **🛠️ Solutions:**

* Impute values (mean/median for numeric, mode for categorical)
* Use separate "missing" categories or indicators
* Drop rows/columns if missingness is excessive and not informative

### **3. Target Variable Analysis**

#### **✅ Why:**

* Helps understand class imbalance and guides model selection and evaluation metrics.

#### **🔍 What to look for:**

* Highly imbalanced binary targets (e.g., <5% charge-offs)
* Right-skewed continuous targets like ChargeOff\_Balance

#### **🛠️ Solutions:**

* Use metrics like AUC, F1, or precision-recall for imbalanced classification
* Apply log transformation to right-skewed continuous targets

### **4. Univariate Feature Distributions**

#### **✅ Why:**

* Understand distribution, skewness, and outliers for each feature.

#### **🔍 What to look for:**

* Skewed distributions (common in income/debt)
* Zero-inflated features (many zeros)
* Outliers or spikes at extreme values

#### **🛠️ Solutions:**

* Log/Box-Cox transform skewed features
* Cap or remove extreme outliers (e.g., top 1%)
* Create bins or flag zero-heavy columns

### **5. Bivariate Analysis**

#### **✅ Why:**

* Helps find how features relate to target and whether they are predictive.

#### **🔍 What to look for:**

* Strong differences between charge-off and non-charge-off groups
* Features with little separation (low predictive value)

#### **🛠️ Solutions:**

* Keep features with good separation
* Remove or de-prioritize low-variance or weakly predictive variables

### **6. Correlation Analysis**

#### **✅ Why:**

* Identify features most correlated with targets or each other (multicollinearity).

#### **🔍 What to look for:**

* High correlation with target → potentially useful feature
* High correlation between predictors → may cause multicollinearity

#### **🛠️ Solutions:**

* Drop redundant features (correlation > 0.9)
* Use PCA or regularization (Ridge/Lasso) to reduce dimensionality

### **7. Feature Engineering Opportunities**

#### **✅ Why:**

* Helps you extract more signal from existing data.

#### **🔍 What to create:**

* Ratios like Unsecured\_Debt / Income (Debt-to-Income)
* Binning of FICO into ranges (e.g., “Poor”, “Fair”)
* Flags (e.g., No\_Hit + FICO is null → true new-to-credit indicator)

#### **🛠️ Solutions:**

* Normalize or scale new features  
  + ChargeOff\_Balance used to predict Ever\_ChargeOff → invalid
  + Delinquency\_90\_days\_last\_12mo might indirectly capture a charge-off

#### **🛠️ Solutions:**

* Carefully exclude leaked features
* Apply proper feature filtering and temporal logic