# Coupon Usage Prediction Model

## Machine Learning for Customer Transaction Analysis

Name Goes Here

University Goes Here

February 18, 2026

# Problem Statement

**Research Question:** Do transaction patterns and discount behaviors predict coupon usage?
**Goal:** Improve coupon targeting and reduce marketing waste
**Target Variable:**

- `coupon_used` (1 = used, 0 = not used) - derived from coupon usage records

**Problem Type:** Binary classification

- Features: transaction amounts, customer behavior, time patterns
- Response: whether a customer will use a coupon in a transaction

# Data Description

**Data Sources:**

| Dataset | Records | Description |
|---|---|---|
| Wallet Transactions | 490,942 | Customer purchases |
| Coupon Usage | 75,676 | Coupons actually used |
| Coupon Distribution | 211,712 | Coupons sent to customers |

**Key Patterns to Explore:**

- Relationship between transaction amount and coupon usage
- Impact of discount depth on redemption rates
- Time-based patterns (hour, day of week)
- Customer behavior history as predictor

# Variable Types

**Numeric (8):**

- `tran_amt`, `discounts_amt`, `point_amt`
- `coupon_used_count`, `total_coupon_used_amt`
- `coupon_send_count`, `total_coupon_send_amt`
- `benefit_ratio`, `discount_ratio`, `savings_pct`

**Categorical (5):**

- `station_code`
- `attributionorgcode`
- `transactionorgcode`
- `hour`, `day_of_week`
- `is_weekend`, `is_morning`

**Target Variable:** `coupon_used` (Binary: 1 = Used, 0 = Not Used)

# Data Preprocessing I

1. **Feature Selection (Removed IDs/Hashes):**
   - membercode, order_no, external_order_no
   - coupon_code, user_id
2. **Outlier Removal (IQR Method):**
   - Method: Removed values outside 5th-95th percentile (IQR-based)
   - Applied to: tran_amt, discounts_amt, point_amt
   - Why: Extreme values distort model training and visualization
   - Records reduced: 490,942 $\rightarrow$ 421,590 ( 14% removed)
3. **Colinearity Removal:**
   - receivable_amt (99% correlated with tran_amt)
   - net_amount, total_benefit (redundant)

# Features I

**Remaining Features (15):**

- Transaction: `tran_amt`, `discounts_amt`, `point_amt`
- Customer/Store: `station_code`, `attributionorgcode`, `transactionorgcode`
- Time-based: `hour`, `day_of_week`, `is_weekend`, `is_morning`, etc.
- Aggregated: `coupon_used_count`, `total_coupon_used_amt`
- `coupon_send_count`, `total_coupon_send_amt`

**Engineered Features (5):**

- `benefit_ratio`, `discount_ratio`
- `savings_pct`, `point_to_discount_ratio`
- `tran_to_receivable_ratio`

# Feature Selection

**Removed Irrelevant Features (IDs and Hashes):**

- `membercode` - Customer ID (hash)
- `order_no` - Transaction ID (hash, no date pattern)
- `external_order_no` - External ID (hash)
- `coupon_code` - Coupon ID (hash)
- `user_id` - User ID (hash)

**Impact:** Removing `membercode` improved accuracy from 64% to 66%

# Key Libraries

- **pandas**: Data manipulation and analysis
- **numpy**: Numerical computations
- **sklearn**: Machine learning models and evaluation
- **tensorflow**: Neural network implementation
- **matplotlib**, **seaborn**: Data visualization

# Sample Data: Raw Input

| membercode | tran_amt | discounts_amt | coupon_used |
|---|---|---|---|
| M001 | 150.00 | 20.00 | 1 |
| M002 | 85.50 | 10.00 | 0 |
| M003 | 220.00 | 35.00 | 1 |
| M004 | 60.00 | 5.00 | 0 |
| M005 | 180.00 | 25.00 | 1 |

# Sample Data: Engineered Features

| tran_amt | discounts_amt | benefit_ratio | discount_ratio | savings_pct |
|---|---|---|---|---|
| 150.00 | 20.00 | 0.133 | 0.133 | 13.3% |
| 85.50 | 10.00 | 0.117 | 0.117 | 11.7% |
| 220.00 | 35.00 | 0.159 | 0.159 | 15.9% |
| 60.00 | 5.00 | 0.083 | 0.083 | 8.3% |
| 180.00 | 25.00 | 0.139 | 0.139 | 13.9% |

# Key Functions

**Data Processing:**

- `pd.read_csv()`: Load datasets
- `df.merge()`: Join datasets
- `df.drop()`: Remove columns
- `df.quantile()`: Outlier removal

**Modeling:**

- `train_test_split()`: Split data
- `GradientBoostingClassifier()`: Build model
- `cross_val_score()`: Validate
- `feature_importances_`: Get importance

# Model Architecture I

- **Algorithms**:
  - Random Forest Classifier
  - Support Vector Machine (SVM) with RBF kernel
  - Gradient Boosting Classifier
  - Logistic Regression (AUC-based selection)
  - Neural Network (3 hidden layers: 256-128-64)
- **Random Forest Parameters**:
  - 100 estimators
  - Max depth: 20
- **Gradient Boosting Parameters**:
  - 100 iterations
  - Max depth: 10
  - Learning rate: 0.1
- **Logistic Regression**:
  - Balanced class weights

# Model Architecture II

- AUC-based model selection
- **Validation**: 3-Fold Cross Validation
- **Sample Size**: 20,000 records

# Neural Network: Class Weights

**Impact of Class Weights on Neural Network:**

**Naive (Unbalanced) Weights:** $\{0{:}1,\ 1{:}2\}$

- Accuracy: 62.92%
- Recall: **92.06%**
- Precision: 49.59%
- Result: Predicts almost everything as positive

**Balanced Weights:** $\{0{:}1,\ 1{:}1\}$

- Accuracy: **71.85%**
- Recall: 58.11%
- Precision: **62.29%**
- Result: Better balance between precision and recall

Key insight: Naive weights over-predict positives, achieving high recall but low precision. Balanced weights improve overall accuracy and create a more useful model.

# Model Performance (20,000 sample)

| Metric | Random Forest | SVM | Gradient Boost | Logistic Reg | Neural Net |
|---|---|---|---|---|---|
| Accuracy | **74.20%** | 69.83% | 73.78% | 62.15% | 71.85% |
| AUC | **0.826** | 0.749 | 0.818 | 0.687 | 0.787 |
| Precision | 64.66% | 61.10% | 64.27% | 48.64% | 62.29% |
| Recall | 64.75% | 47.84% | 63.52% | 65.09% | 58.11% |
| F1 Score | **64.71%** | 53.67% | 63.89% | 55.68% | 60.13% |

Random Forest achieves best accuracy and AUC. Neural Network improved with balanced class weights.

# Sample Size Comparison

| Model | 20,000 Accuracy | 421,590 Accuracy | Improvement |
|---|---|---|---|
| Random Forest | 74.22% | **75.60%** | +1.38% |
| Gradient Boosting | 73.62% | **74.13%** | +0.51% |

Larger sample size improves model accuracy, with Random Forest showing greater benefit from more data.

# Confusion Matrices I

**Random Forest (Accuracy: 74.22%):**

$$\begin{bmatrix} \sim 2100 & \sim 440 \\ \sim 580 & \sim 880 \end{bmatrix}$$

| | Predicted: No | Predicted: Yes |
|---|---|---|
| Actual: No | 2100 (TN) | 440 (FP) |
| Actual: Yes | 580 (FN) | 880 (TP) |

# Confusion Matrices II

**Gradient Boosting (Accuracy: 73.62%):**

$$\begin{bmatrix} \sim 2080 & \sim 460 \\ \sim 590 & \sim 870 \end{bmatrix}$$

| | Predicted: No | Predicted: Yes |
|---|---|---|
| Actual: No | 2080 (TN) | 460 (FP) |
| Actual: Yes | 590 (FN) | 870 (TP) |

# Confusion Matrices III

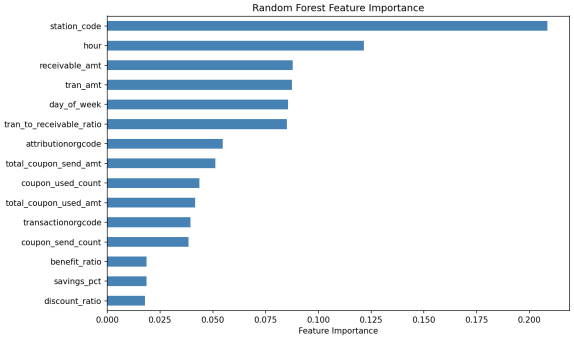
**Neural Network (Accuracy: 71.85%, Recall: 58.11%):**

$$\begin{bmatrix} 2025 & 514 \\ 612 & 849 \end{bmatrix}$$

| | Predicted: No | Predicted: Yes |
|---|---|---|
| Actual: No | 2025 (TN) | 514 (FP) |
| Actual: Yes | 612 (FN) | 849 (TP) |

Neural Network improved with balanced class weights.

# Feature Importance



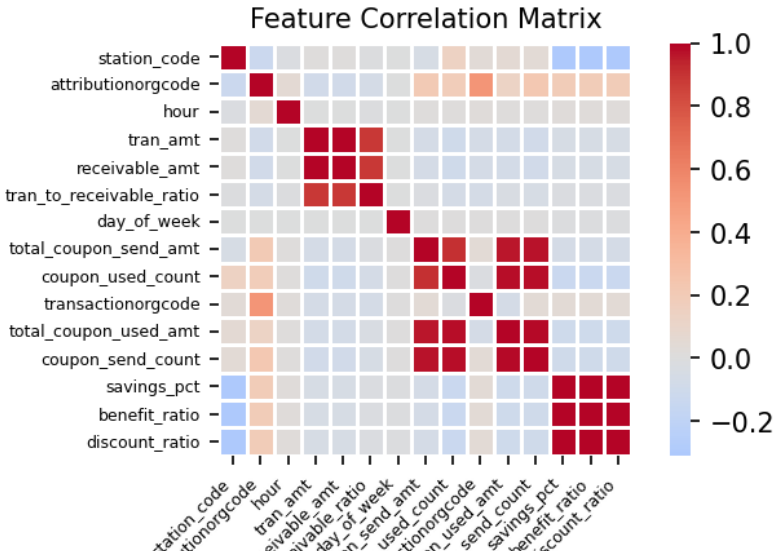

Top Features:

1. `station_code` (23.5%)
2. `hour` (9.45%)
3. `receivable_amt` (8.36%)
4. `tran_to_receivable_ratio` (8.29%)
5. `tran_amt` (8.15%)
6. `day_of_week` (7.44%)
7. `attributionorgcode` (5.45%)
8. `total_coupon_send_amt` (5.17%)
9. `total_coupon_used_amt` (4.81%)
10. `coupon_used_count` (4.08%)

**How Feature Importance was Derived:**

- Using Random Forest classifier's built-in feature importance
- Measures mean decrease in impurity (Gini importance)
- Averaged across all 100 decision trees
- Higher value = more predictive power

# Correlation Matrix Heatmap



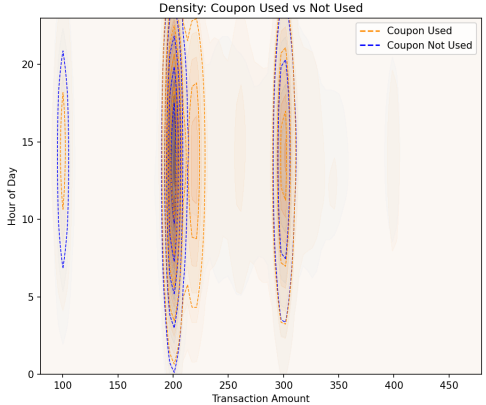
# Dimensionality Reduction

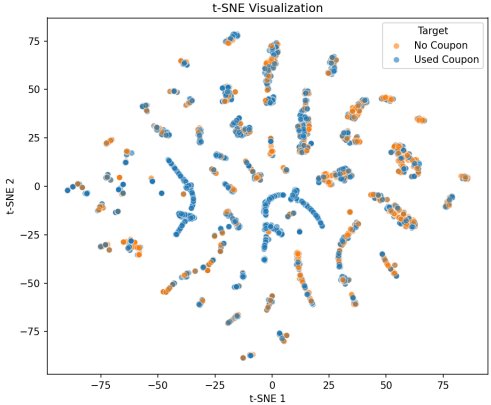
**Density Comparison:**

- Kernel density overlay of transaction amount vs hour
- Shows where each class concentrates

**t-SNE Visualization:**

- Non-linear embedding
- Preserves local structure



# PCA Visualization

# Conclusions I

- Random Forest achieves **74.20% accuracy** with **0.826 AUC** - best model
- Gradient Boosting: 73.78% accuracy, 0.818 AUC
- Neural Network: 71.85% accuracy with balanced class weights (3 layers: 256-128-64)
- Feature selection: reduced from 23 to 15 features without accuracy loss
- Removed zero-importance features: `point_amt`, `point_to_discount_ratio`
- Top features: station_code (23.5%), attributionorgcode (9.4%)
- 3-Fold Cross Validation confirms model stability
- Future work: XGBoost/LightGBM, more feature engineering