# Classwork 2: Feature Engineering

Building Timeline-Compliant Features from Transaction Data

Prof. Asc. Endri Raco, Ph.D.

November 2025

## Overview

**Objective:** Apply feature engineering techniques to real donor transaction data to build predictive variables that respect the temporal timeline.

**Dataset:** `donor_transactions.csv` (provided on course portal)

**Submission:** R Markdown file (.Rmd) + knitted PDF

**Due Date:** November 20, 2025, 23:59

**Weight:** 15% of final grade

---

## Part 1: Data Understanding (15 points)

### Task 1.1: Load and Explore Data (5 points)

Load the provided dataset and answer the following:

```r
# Load required libraries
library(tidyverse)
library(lubridate)

# Load data
donors <- read_csv("donor_transactions.csv")

# Your code here
```

**Questions to answer:**

a) How many unique donors are in the dataset?
b) What is the date range of transactions?
c) How many transactions per donor (mean, median, min, max)?
d) What is the distribution of donation amounts?

**Deliverable:** Summary statistics table and at least 2 visualizations.

---

### Task 1.2: Identify Data Quality Issues (5 points)

Examine the data for potential problems:

```r
# Check for missing values
summary(donors)

# Check for duplicates
# Check for invalid dates
# Check for negative amounts
```

**Questions:**

a) Are there any missing values? Which columns?
b) Are there any duplicate transactions?
c) Are there any transactions with amount = 0 or negative?
d) Are dates in correct chronological order?

**Deliverable:** Written report of issues found and proposed handling strategy.

---

### Task 1.3: Define Reference Date (5 points)

Set the reference date and create the temporal split:

```r
# Define reference date (use 2024-01-01)
reference_date <- as.Date("2024-01-01")

# Filter observation period data
observation_data <- donors %>%
  filter(date < reference_date)

# Filter target period data (next 30 days)
target_data <- donors %>%
  filter(date >= reference_date,
         date < reference_date + days(30))
```

**Questions:**

a) How many transactions are in the observation period?
b) How many transactions are in the target period?
c) How many donors gave in the target period?

**Deliverable:** Summary counts and verification that no future data leaks into features.

---

## Part 2: RFM Features (25 points)

### Task 2.1: Calculate Recency (8 points)

Create recency features:

```r
# Calculate days since last donation for each donor
recency_features <- observation_data %>%
  group_by(donor_id) %>%
  summarise(
    last_donation_date = max(date),
    days_since_last = as.numeric(reference_date - last_donation_date)
  )
```

```r
# Create recency categories
recency_features <- recency_features %>%
  mutate(
    recency_segment = case_when(
      days_since_last <= 30 ~ "Active",
      days_since_last <= 90 ~ "Warm",
      days_since_last <= 365 ~ "Cooling",
      TRUE ~ "Cold"
    )
  )
```

**Deliverable:**

- Table showing distribution of donors across recency segments
- Visualization (bar chart or histogram)
- Brief interpretation: Which segment has the most donors?

---

### Task 2.2: Calculate Frequency (8 points)

Create frequency features for the 12-month window before reference date:

```r
# Define 12-month window
window_12m_start <- reference_date - months(12)

# Calculate frequency metrics
frequency_features <- observation_data %>%
  filter(date >= window_12m_start) %>%
  group_by(donor_id) %>%
  summarise(
    donation_count = n(),
    unique_months = n_distinct(floor_date(date, "month")),
    avg_days_between = # Calculate this
  )
```

**Tasks:**

a) Complete the code to calculate average days between donations
b) Create a `frequency_segment` variable with categories: "Monthly" (12+ donations), "Regular" (4-11), "Occasional" (2-3), "Rare" (1)
c) Calculate a "regularity score" = unique_months / 12

**Deliverable:**

- Frequency distribution table
- Scatter plot: donation_count vs. regularity score
- Interpretation of correlation

---

### Task 2.3: Calculate Monetary Value (9 points)

Create monetary features:

```r
# Calculate monetary metrics
monetary_features <- observation_data %>%
  filter(date >= window_12m_start) %>%
  group_by(donor_id) %>%
```

```r
  summarise(
    total_value_12m = sum(amount),
    avg_donation = mean(amount),
    max_donation = max(amount),
    min_donation = min(amount),
    std_donation = sd(amount)
  ) %>%
  mutate(
    cv_donation = std_donation / avg_donation,  # Coefficient of variation
    value_tier = # Create categories here
  )
```

**Tasks:**

a) Create `value_tier` with categories based on total_value_12m:
   - "Major" (1000+)
   - "Premium" (500-999)
   - "Standard" (100-499)
   - "Entry" (<100)

b) Identify donors with high CV (>1.0) - what does this mean?

**Deliverable:**

- Summary statistics table by value_tier
- Box plot: distribution of avg_donation by value_tier
- List top 10 donors by total_value_12m

---

# Part 3: Trend Features (25 points)

## Task 3.1: Calculate Short-Term Trends (10 points)

Compare recent 3 months to previous 3 months:

```r
# Recent period: 3 months before reference date
recent_start <- reference_date - months(3)
recent_end <- reference_date

# Previous period: 3-6 months before reference date
previous_start <- reference_date - months(6)
previous_end <- reference_date - months(3)

# Aggregate recent period
recent_agg <- observation_data %>%
  filter(date >= recent_start, date < recent_end) %>%
  group_by(donor_id) %>%
  summarise(
    sum_recent = sum(amount),
    count_recent = n()
  )

# Aggregate previous period
previous_agg <- observation_data %>%
  filter(date >= previous_start, date < previous_end) %>%
  group_by(donor_id) %>%
```

```
  summarise(
    sum_previous = sum(amount),
    count_previous = n()
  )

# Calculate trends
trends <- recent_agg %>%
  full_join(previous_agg, by = "donor_id") %>%
  mutate(
    sum_recent = replace_na(sum_recent, 0),
    sum_previous = replace_na(sum_previous, 0),

    # Your calculations here
    value_change = # Absolute change
    pct_change = # Percentage change
    trend_direction = # Categorize as "Increasing", "Stable", "Decreasing"
  )
```

**Deliverable:**

- Complete the trend calculations
- Distribution of trend_direction (count and %)
- Identify 5 donors with strongest growth and 5 with steepest decline

---

## Task 3.2: Calculate Long-Term Trends (8 points)

Compare last 6 months to previous 6 months (within 12-month window):

```
# Period 1: Months 7-12 before reference date
# Period 2: Months 1-6 before reference date

# Calculate trend and momentum indicators
```

**Tasks:**

a) Calculate the same metrics as Task 3.1 but for 6-month periods
b) Create a `momentum` variable that combines both trend periods:
   - "Accelerating" if both periods show growth
   - "Decelerating" if growth is slowing
   - "Declining" if both periods show decline
   - "Recovering" if decline is reversing

**Deliverable:**

- Momentum distribution table
- Compare donors with "Accelerating" vs. "Declining" momentum on their RFM scores

---

## Task 3.3: Growth Rate Calculation (7 points)

Calculate compound growth rate across quarters:

```
# Get quarterly totals for each donor
quarterly_data <- observation_data %>%
  filter(date >= reference_date - months(12)) %>%
  mutate(quarter = floor_date(date, "quarter")) %>%
```

```r
  group_by(donor_id, quarter) %>%
  summarise(quarterly_total = sum(amount), .groups = "drop")

# Calculate growth rate for donors with 4+ quarters
# CAGR formula: (End/Start)^(1/n) - 1
```

**Deliverable:**

- Calculate CAGR for donors with complete quarterly data
- Histogram of growth rates
- Identify donors with growth rate > 50% (high potential!)

---

# Part 4: Advanced Features (20 points)

## Task 4.1: Ratio Features (7 points)

Create ratio-based features:

```r
# Join all feature sets
feature_set <- basetable %>%
  left_join(recency_features, by = "donor_id") %>%
  left_join(frequency_features, by = "donor_id") %>%
  left_join(monetary_features, by = "donor_id") %>%
  left_join(trends, by = "donor_id")

# Calculate ratios
feature_set <- feature_set %>%
  mutate(
    # Recent activity as proportion of total
    ratio_3m_to_12m = sum_recent / (total_value_12m + 0.01),

    # Max gift concentration
    max_concentration = max_donation / (total_value_12m + 0.01),

    # Average gift relative to lifetime
    # Add more ratios
  )
```

**Deliverable:**

- Calculate at least 5 meaningful ratio features
- Create summary statistics for each ratio
- Interpret what each ratio tells you about donor behavior

---

## Task 4.2: Interaction Features (7 points)

Create interaction terms combining RFM components:

```r
# RFM Interactions
feature_set <- feature_set %>%
  mutate(
    # Recency-Frequency interaction
    RF_score = (1 / (days_since_last + 1)) * donation_count,
```

```r
    # Frequency-Monetary interaction
    FM_score = donation_count * avg_donation,

    # Your additional interactions
  )
```

**Tasks:**

a) Create at least 3 interaction features
b) Calculate correlation between interactions and individual components
c) Identify donors with highest RF_score

**Deliverable:**

- Correlation matrix (heatmap visualization)
- Top 20 donors ranked by RF_score
- Interpretation: Why might interactions be more predictive than individual features?

---

## Task 4.3: Missing Value Handling (6 points)

Handle missing values appropriately:

```r
# Identify patterns of missingness
missing_summary <- feature_set %>%
  summarise(across(everything(), ~sum(is.na(.))))

# Create missingness flags
feature_set <- feature_set %>%
  mutate(
    flag_no_activity_3m = as.integer(is.na(sum_recent) | sum_recent == 0),
    flag_no_activity_12m = as.integer(is.na(total_value_12m) | total_value_12m == 0),
    # Add more flags
  )

# Impute missing values
feature_set <- feature_set %>%
  mutate(
    # Strategy for different variable types
    days_since_last = replace_na(days_since_last, 9999),
    total_value_12m = replace_na(total_value_12m, 0),
    # Continue for all variables
  )
```

**Deliverable:**

- Missing value report (which variables, how many, why?)
- Imputation strategy documented for each variable type
- Justification for your choices

---

# Part 5: Feature Analysis (15 points)

## Task 5.1: Create Target Variable (5 points)

Define the prediction target:

```r
# Create target: Did donor give in next 30 days?
target <- target_data %>%
  group_by(donor_id) %>%
  summarise(
    donated_in_target = 1,
    target_amount = sum(amount)
  )

# Join to feature set
final_dataset <- feature_set %>%
  left_join(target, by = "donor_id") %>%
  mutate(
    donated_in_target = replace_na(donated_in_target, 0)
  )
```

**Questions:**

a) What is the baseline conversion rate (% who donated)?
b) How does this compare to industry standards (typically 3-8%)?

**Deliverable:** Target variable statistics and distribution

---

## Task 5.2: Feature-Target Relationships (5 points)

Explore which features correlate with the target:

```r
# For continuous features
continuous_features <- c("days_since_last", "donation_count",
                         "total_value_12m", "RF_score")

correlations <- final_dataset %>%
  select(all_of(continuous_features), donated_in_target) %>%
  cor(use = "complete.obs")

# For categorical features
categorical_features <- c("recency_segment", "frequency_segment",
                          "value_tier", "trend_direction")

# Calculate conversion rate by segment
```

**Deliverable:**

- Correlation matrix for continuous features
- Conversion rate tables for categorical features (with bar charts)
- Identify top 5 most predictive features

---

## Task 5.3: Feature Documentation (5 points)

Create a feature catalog:

```r
feature_catalog <- tibble(
  feature_name = c("days_since_last", "donation_count",
                   "RF_score", "ratio_3m_to_12m", "..."),
  feature_type = c("Numeric", "Numeric", "Numeric",
                   "Ratio", "..."),
  calculation_window = c("Point-in-time", "12 months",
                         "Derived", "3m vs 12m", "..."),
  business_meaning = c("Recency of last donation",
                       "Frequency of giving",
                       "Combined engagement score",
                       "Recent behavior vs. average",
                       "..."),
  missing_handling = c("Set to 9999 for inactive",
                       "Set to 0 for no activity",
                       "Requires both R and F",
                       "NA if denominator zero",
                       "...")
)
```

**Deliverable:**

- Complete feature catalog for ALL features created (minimum 15 features)
- Well-formatted table (use `kable` or `gt` package)

---

# Bonus Tasks (10 points extra credit)

## Bonus 1: Seasonality Features (5 points)

Calculate seasonal patterns:

```r
# Extract month from dates
# Calculate average donation by month for each donor
# Create seasonal index (ratio to annual average)
# Add current month's index to feature set
```

## Bonus 2: Advanced Visualization (5 points)

Create an interactive dashboard using `plotly` or `shiny` showing:

- RFM segment distribution
- Trend analysis over time
- Feature importance visualization

---

# Grading Rubric

| Component | Points | Criteria |
|---|---|---|
| Part 1: Data Understanding | 15 | Correct calculations, clear visualizations, thorough data quality check |
| Part 2: RFM Features | 25 | Accurate RFM calculations, proper categorization, good interpretation |

| Component | Points | Criteria |
|---|---|---|
| Part 3: Trend Features | 25 | Correct trend calculations, meaningful momentum indicators |
| Part 4: Advanced Features | 20 | Creative ratio/interaction features, proper missing value handling |
| Part 5: Feature Analysis | 15 | Clear target definition, insightful feature-target analysis, complete documentation |
| **Code Quality** | **10** | **Clean, commented, reproducible code** |
| **Report Quality** | **10** | **Clear explanations, professional formatting, no errors** |
| Bonus | +10 | Extra credit for going beyond requirements |
| **Total** | **120** | **(100 + 20 bonus)** |

---

# Submission Guidelines

## What to Submit

1. **R Markdown file** (.Rmd) with all code and explanations
2. **Knitted PDF** showing all output
3. **Feature catalog** (can be embedded in PDF or separate CSV)

## File Naming Convention

`Classwork2_FirstnameLastname.Rmd`

`Classwork2_FirstnameLastname.pdf`

## Submission Platform

Upload to course portal under "Assignments > Classwork 2"

---

# Tips for Success

1. **Start early** - Feature engineering takes time to get right
2. **Document everything** - Explain your reasoning for each feature
3. **Validate timeline** - Double-check no future data leakage
4. **Test incrementally** - Run code frequently to catch errors
5. **Visualize** - Charts often reveal insights that numbers hide
6. **Think business** - Features should make sense to non-technical stakeholders
7. **Handle edge cases** - What if donor has no history? One transaction?
8. **Use meaningful names** - `days_since_last` not `var_x47`

---

## Frequently Asked Questions

**Q: What if I find more data quality issues than listed?**

A: Document all issues you find. This shows attention to detail!

**Q: Can I create features not mentioned in the assignment?**

A: Absolutely! Creativity is encouraged. Just document your reasoning.

**Q: How many features should I create total?**

A: Minimum 15 required. Top students typically create 20-25.

**Q: What if a donor has zero transactions in the observation period?**

A: They should still be in your basetable with appropriate defaults (e.g., days_since_last = 9999, total_value = 0)

**Q: Can I use external packages beyond tidyverse?**

A: Yes, but document which packages and why.

**Q: Should I build a model?**

A: Not required for this assignment. Focus on feature engineering quality.

---

## Academic Integrity

- You may discuss concepts with classmates
- You must write your own code independently
- Copy-paste from classmates = automatic zero
- Using ChatGPT for debugging = OK
- Using ChatGPT to write entire solution = NOT OK

**Remember:** The goal is to learn, not just to complete the assignment.

---

## Good Luck!

Remember: Better features make better predictions.

Spend time understanding the data before engineering features.