

Exercise.

You are part of a data science team at a subscription-based company (e.g., streaming service or SaaS platform). Your goal is to predict **customer churn** (whether a customer will leave the service or continue their subscription) using historical customer data.

Dataset Overview

You are provided with a dataset containing the following features:

Feature	Description
<code>customer_id</code>	Unique identifier for each customer
<code>subscription_length</code>	Length of the subscription (in months)
<code>last_purchase</code>	Date of the most recent purchase
<code>total_spent</code>	Total amount the customer has spent
<code>churn</code>	Binary target variable (1 = churned, 0 = active)

Exercise Tasks

Part 1 — Data Understanding

1. What kind of data preprocessing steps would you perform before training a Random Forest model?
(Hint: Consider data types, missing values, feature engineering from `last_purchase`, and potential scaling or encoding needs.)
2. Suggest at least two new features you could derive that might improve the predictive power of the model.
(For example, days since last purchase, average monthly spend, etc.)

Part 2 — Model Building

3. Explain why **Random Forest** is a good choice for this problem compared to a single decision tree.
(Discuss bias-variance tradeoff, robustness, and interpretability.)
4. Outline the key hyperparameters of a Random Forest classifier that you would tune and explain what each one controls.
(Examples: `n_estimators`, `max_depth`, `min_samples_split`, `max_features`.)

Part 3 — Model Evaluation

5. What metrics would you use to evaluate your churn prediction model and why? (Consider that churn is often imbalanced)
6. Suppose the model shows high accuracy but low recall for the churned customers. What does this indicate, and what could you do to fix it? (Hint: class weighting, threshold adjustment, sampling strategies.)

Part 4 — Interpretation and Actionability

7. After training your Random Forest model, you obtain feature importances.
 - o `total_spent`: 0.45
 - o `subscription_length`: 0.30
 - o `days_since_last_purchase`: 0.20
 - o `other features`: 0.05

How would you interpret these results, and what business insights can you derive?

8. Describe one actionable recommendation your company could take based on the model's output. (Example: targeted retention campaigns for customers with low total spending and long inactivity periods.)
9. Decision trees in a Random Forest use *entropy* to measure how mixed or uncertain the data is at each split. Explain what entropy represents in the context of predicting customer churn. How does information gain guide the model in choosing the best feature to split on? In real-world terms, what does a high information gain tell you about a feature like `subscription_length` or `days_since_last_purchase` when predicting whether a customer will churn? (Hint: Think about how much a feature reduces uncertainty about customer behavior.)

Part 1 — Data Understanding

1. Data Preprocessing Steps

Before training a **Random Forest** model, we should perform the following preprocessing steps:

- **Handle data types**
 - Ensure `customer_id` is treated as an **identifier** (drop it from model features).
 - Convert `last_purchase` to a **datetime** object for feature engineering.
- **Handle missing values**
 - If `subscription_length` or `total_spent` contain missing values, consider:
 - Filling with median or mean (for numerical features).
 - Dropping rows if missingness is small and random.
 - If `last_purchase` is missing, it may indicate **inactivity**, which is meaningful — create an indicator feature (`no_recent_purchase = 1`).
- **Feature engineering from `last_purchase`**
 - Compute `days_since_last_purchase = today's_date - last_purchase`.
 - This converts a date into a meaningful numerical feature correlated with churn.
- **Encoding and scaling**
 - Random Forests do **not require feature scaling** (they are based on splits, not distances).
 - If categorical variables existed, encode them with label encoding.
- **Train-test split**
 - Split the dataset into **train (70–80%)** and **test (20–30%)**, ensuring random stratification on `churn`.

2. New Feature Ideas

You could engineer new features that capture behavioral and temporal dynamics:

1. **Days since last purchase**
 - `days_since_last_purchase = (reference_date - last_purchase).days`
 - Indicates recency of activity; higher values may correlate with higher churn probability.
2. **Average monthly spend**
 - `avg_monthly_spend = total_spent / subscription_length`
 - Measures engagement value and spending intensity.

Other possible features:

- **spending_trend** (increase/decrease over time if time series data available)
- **purchase_frequency** (number of purchases per month)
- **tenure_category** (bucketed subscription length)

Part 2 — Model Building

3. Why Random Forest is a Good Choice

Random Forest (RF) is preferable to a single decision tree because:

- **Reduces overfitting (bias–variance tradeoff):**
Individual trees have high variance, but RF averages many trees built on random subsets, reducing overfitting and improving generalization.
- **Robustness:**
Less sensitive to noisy data and outliers due to averaging across multiple models.
- **Handles mixed data types:**
Works well with both categorical and numerical features without much preprocessing.
- **Interpretability:**
Still provides **feature importances**, offering business insights (unlike many “black-box” models).

4. Key Hyperparameters to Tune

Hyperparameter	Description	Effect
n_estimators	Number of trees in the forest	More trees → more stable predictions but higher computation time
max_depth	Maximum depth of each tree	Controls model complexity; deeper trees may overfit
min_samples_split	Minimum number of samples required to split an internal node	Larger values → simpler trees (reduces overfitting)
min_samples_leaf	Minimum number of samples at a leaf node	Ensures leaves have enough data points
max_features	Number of features considered for best split	Adds randomness; helps decorrelate trees
class_weight	Weight assigned to each class	Helps handle class imbalance (important for churn prediction)

Part 3 — Model Evaluation

5. Evaluation Metrics

Since **churn prediction** is usually **imbalanced** (few churners vs. many active users):

- **Precision:** Of the customers predicted to churn, how many actually churned?
- **Recall (Sensitivity):** Of the actual churners, how many were correctly identified?
- **F1-score:** Harmonic mean of precision and recall — balances both.

- **Accuracy alone is misleading** — a model could predict all customers as “not churned” and still achieve high accuracy.

6. High Accuracy but Low Recall for Churners

Interpretation:

- The model correctly identifies most active customers but **misses many actual churners**.
- It’s conservative — prefers predicting “no churn.”

Fixes:

- **Class weighting:** Use `class_weight='balanced'` in RandomForest to penalize misclassification of minority (churn) class.
- **Sampling strategies:**
 - **Oversample** churners.
 - **Undersample** non-churners.
- **Threshold tuning:** Lower the decision threshold (e.g., from 0.5 to 0.3) to increase recall at the expense of precision.
- **Cost-sensitive learning:** Penalize false negatives more heavily.

Part 4 — Interpretation and Actionability

7. Interpreting Feature Importances

Feature	Importance	Interpretation
total_spent (0.45)	Most influential feature	Customers who spend less overall are more likely to churn — spending is a strong loyalty indicator.
subscription_length (0.30)	Second most important	Longer-tenure customers are less likely to churn; short-term subscribers are riskier.
days_since_last_purchase (0.20)	Third most important	Recency matters — long inactivity periods signal disengagement.
Other features (0.05)	Minor impact	Have limited predictive value.

Business Insight:

Retention is most influenced by **customer value (total_spent)** and **engagement recency**. Focus retention efforts on **low-spending, recently inactive** users.

8. Actionable Recommendation

Based on the model:

- **Targeted retention campaigns:**
Offer personalized discounts or renewal incentives to customers with **low total spending** and **high days since last purchase**.
- Example:

Send an automated “We miss you” email or a **special offer** after 30 days of inactivity to low-value customers.

This helps **reduce churn** and **increase customer lifetime value**.

9. Entropy and Information Gain in Churn Prediction

- **Entropy** measures **uncertainty** or “mixing” of churn and non-churn cases in a node.
 - High entropy → data is mixed (50% churn, 50% active).
 - Low entropy → data is pure (mostly churn or mostly active).
- **Information gain** = reduction in entropy after splitting on a feature.
 - The feature that gives the **largest reduction in uncertainty** (highest information gain) is chosen for the split.

Example (real-world meaning):

- If splitting on `subscription_length` separates most churners (short-term users) from loyal users (long-term), it has **high information gain**.
- A **high information gain** for `days_since_last_purchase` means that knowing how long it’s been since a purchase **greatly reduces uncertainty** about churn likelihood.

Interpretation:

Features with high information gain are those that **best explain customer behavior** — they provide