M504D

AI and Applications

Individual Final Project

Student Name: Anh Tuan Ta
Student ID: GH1046139.
Student Email: anh.ta@gisma-student.com
LinkedIn: https://www.linkedin.com/in/ta-anh-tuan-ai-engineer

MSc of DATA SCIENCE, AI AND DIGITAL BUSINESS

SaaS Customer Churn Data Science Consulting

Business Context - SaaS Customer Churn

Role: Data Science Consultant
Client: A B2B SaaS company aiming to reduce churn and increase revenue retention.
Data: `saas-customer-churn.csv` — customer profile, plan, product usage/engagement, support interactions, billing/revenue, tenure, and churn labels.

Why this analysis?
Churn erodes recurring revenue and increases acquisition costs. We'll explore data quality, engineer useful features, and answer key business questions (churn drivers, risky segments, revenue at risk, retention trends). We'll also fit a lightweight baseline model to quantify feature effects.

Deliverables:

- Data audit (quality, schema, missingness)
- Preprocessing (types, nulls, outliers, features)
- 8–10 business questions answered with code + commentary
- Final insights & data-driven recommendations

Dataset Link:

- https://drive.google.com/drive/folders/1Iww7mcD9RTmdZz1xhP2uJ5AG86R7fyWD?usp=drive_link

1. Installing Libraries

In [1]:
```
import os, sys, pathlib
print("Python:", sys.executable)
print("sys.prefix:", sys.prefix)
env_name = os.environ.get("CONDA_DEFAULT_ENV") or pathlib.Path(sys.prefix
print("Conda env:", env_name)
```

```
Python: /Users/taanhtuan/miniconda3/envs/churn/bin/python3.12
sys.prefix: /Users/taanhtuan/miniconda3/envs/churn
Conda env: churn
```

In [2]:
```
!pip install -r requirements.txt
```

```
Requirement already satisfied: numpy==1.26.4 in /Users/taanhtuan/miniconda
3/envs/churn/lib/python3.12/site-packages (from -r requirements.txt (line
1)) (1.26.4)
Requirement already satisfied: pandas==2.2.2 in /Users/taanhtuan/miniconda
3/envs/churn/lib/python3.12/site-packages (from -r requirements.txt (line
2)) (2.2.2)
Requirement already satisfied: scipy==1.11.4 in /Users/taanhtuan/miniconda
3/envs/churn/lib/python3.12/site-packages (from -r requirements.txt (line
3)) (1.11.4)
Requirement already satisfied: matplotlib==3.8.4 in /Users/taanhtuan/minic
onda3/envs/churn/lib/python3.12/site-packages (from -r requirements.txt (l
ine 4)) (3.8.4)
Requirement already satisfied: seaborn==0.13.2 in /Users/taanhtuan/minicon
da3/envs/churn/lib/python3.12/site-packages (from -r requirements.txt (lin
e 5)) (0.13.2)
Requirement already satisfied: scikit-learn==1.5.2 in /Users/taanhtuan/min
iconda3/envs/churn/lib/python3.12/site-packages (from -r requirements.txt
(line 6)) (1.5.2)
Requirement already satisfied: shap==0.45.1 in /Users/taanhtuan/miniconda
3/envs/churn/lib/python3.12/site-packages (from -r requirements.txt (line
7)) (0.45.1)
Requirement already satisfied: tqdm==4.66.4 in /Users/taanhtuan/miniconda
3/envs/churn/lib/python3.12/site-packages (from -r requirements.txt (line
8)) (4.66.4)
Requirement already satisfied: notebook==7.2.2 in /Users/taanhtuan/minicon
da3/envs/churn/lib/python3.12/site-packages (from -r requirements.txt (lin
e 9)) (7.2.2)
Requirement already satisfied: ipykernel==6.29.5 in /Users/taanhtuan/minic
onda3/envs/churn/lib/python3.12/site-packages (from -r requirements.txt (l
ine 10)) (6.29.5)
Requirement already satisfied: ipywidgets==8.1.2 in /Users/taanhtuan/minic
onda3/envs/churn/lib/python3.12/site-packages (from -r requirements.txt (l
ine 11)) (8.1.2)
Requirement already satisfied: python-dateutil>=2.8.2 in /Users/taanhtuan/
miniconda3/envs/churn/lib/python3.12/site-packages (from pandas==2.2.2->-r
requirements.txt (line 2)) (2.9.0.post0)
Requirement already satisfied: pytz>=2020.1 in /Users/taanhtuan/miniconda
3/envs/churn/lib/python3.12/site-packages (from pandas==2.2.2->-r requirem
ents.txt (line 2)) (2025.2)
Requirement already satisfied: tzdata>=2022.7 in /Users/taanhtuan/minicond
a3/envs/churn/lib/python3.12/site-packages (from pandas==2.2.2->-r require
ments.txt (line 2)) (2025.2)
Requirement already satisfied: contourpy>=1.0.1 in /Users/taanhtuan/minico
nda3/envs/churn/lib/python3.12/site-packages (from matplotlib==3.8.4->-r r
equirements.txt (line 4)) (1.3.3)
Requirement already satisfied: cycler>=0.10 in /Users/taanhtuan/miniconda
3/envs/churn/lib/python3.12/site-packages (from matplotlib==3.8.4->-r requ
irements.txt (line 4)) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in /Users/taanhtuan/minic
onda3/envs/churn/lib/python3.12/site-packages (from matplotlib==3.8.4->-r
requirements.txt (line 4)) (4.60.0)
Requirement already satisfied: kiwisolver>=1.3.1 in /Users/taanhtuan/minic
onda3/envs/churn/lib/python3.12/site-packages (from matplotlib==3.8.4->-r
requirements.txt (line 4)) (1.4.9)
Requirement already satisfied: packaging>=20.0 in /Users/taanhtuan/minicon
da3/envs/churn/lib/python3.12/site-packages (from matplotlib==3.8.4->-r re
quirements.txt (line 4)) (25.0)
Requirement already satisfied: pillow>=8 in /Users/taanhtuan/miniconda3/en
vs/churn/lib/python3.12/site-packages (from matplotlib==3.8.4->-r requirem
ents.txt (line 4)) (11.3.0)
```

```
Requirement already satisfied: pyparsing>=2.3.1 in /Users/taanhtuan/minico
nda3/envs/churn/lib/python3.12/site-packages (from matplotlib==3.8.4->-r r
equirements.txt (line 4)) (3.2.5)
Requirement already satisfied: joblib>=1.2.0 in /Users/taanhtuan/miniconda
3/envs/churn/lib/python3.12/site-packages (from scikit-learn==1.5.2->-r re
quirements.txt (line 6)) (1.5.2)
Requirement already satisfied: threadpoolctl>=3.1.0 in /Users/taanhtuan/mi
niconda3/envs/churn/lib/python3.12/site-packages (from scikit-learn==1.5.2
->-r requirements.txt (line 6)) (3.6.0)
Requirement already satisfied: slicer==0.0.8 in /Users/taanhtuan/miniconda
3/envs/churn/lib/python3.12/site-packages (from shap==0.45.1->-r requireme
nts.txt (line 7)) (0.0.8)
Requirement already satisfied: numba in /Users/taanhtuan/miniconda3/envs/c
hurn/lib/python3.12/site-packages (from shap==0.45.1->-r requirements.txt
(line 7)) (0.62.0)
Requirement already satisfied: cloudpickle in /Users/taanhtuan/miniconda3/
envs/churn/lib/python3.12/site-packages (from shap==0.45.1->-r requirement
s.txt (line 7)) (3.1.1)
Requirement already satisfied: jupyter-server<3,>=2.4.0 in /Users/taanhtua
n/miniconda3/envs/churn/lib/python3.12/site-packages (from notebook==7.2.2
->-r requirements.txt (line 9)) (2.17.0)
Requirement already satisfied: jupyterlab-server<3,>=2.27.1 in /Users/taan
htuan/miniconda3/envs/churn/lib/python3.12/site-packages (from notebook==
7.2.2->-r requirements.txt (line 9)) (2.27.3)
Requirement already satisfied: jupyterlab<4.3,>=4.2.0 in /Users/taanhtuan/
miniconda3/envs/churn/lib/python3.12/site-packages (from notebook==7.2.2->
-r requirements.txt (line 9)) (4.2.7)
Requirement already satisfied: notebook-shim<0.3,>=0.2 in /Users/taanhtua
n/miniconda3/envs/churn/lib/python3.12/site-packages (from notebook==7.2.2
->-r requirements.txt (line 9)) (0.2.4)
Requirement already satisfied: tornado>=6.2.0 in /Users/taanhtuan/minicond
a3/envs/churn/lib/python3.12/site-packages (from notebook==7.2.2->-r requi
rements.txt (line 9)) (6.5.2)
Requirement already satisfied: appnope in /Users/taanhtuan/miniconda3/env
s/churn/lib/python3.12/site-packages (from ipykernel==6.29.5->-r requireme
nts.txt (line 10)) (0.1.4)
Requirement already satisfied: comm>=0.1.1 in /Users/taanhtuan/miniconda3/
envs/churn/lib/python3.12/site-packages (from ipykernel==6.29.5->-r requir
ements.txt (line 10)) (0.2.3)
Requirement already satisfied: debugpy>=1.6.5 in /Users/taanhtuan/minicond
a3/envs/churn/lib/python3.12/site-packages (from ipykernel==6.29.5->-r req
uirements.txt (line 10)) (1.8.17)
Requirement already satisfied: ipython>=7.23.1 in /Users/taanhtuan/minicon
da3/envs/churn/lib/python3.12/site-packages (from ipykernel==6.29.5->-r re
quirements.txt (line 10)) (9.5.0)
Requirement already satisfied: jupyter-client>=6.1.12 in /Users/taanhtuan/
miniconda3/envs/churn/lib/python3.12/site-packages (from ipykernel==6.29.5
->-r requirements.txt (line 10)) (8.6.3)
Requirement already satisfied: jupyter-core!=5.0.*,>=4.12 in /Users/taanht
uan/miniconda3/envs/churn/lib/python3.12/site-packages (from ipykernel==6.
29.5->-r requirements.txt (line 10)) (5.8.1)
Requirement already satisfied: matplotlib-inline>=0.1 in /Users/taanhtuan/
miniconda3/envs/churn/lib/python3.12/site-packages (from ipykernel==6.29.5
->-r requirements.txt (line 10)) (0.1.7)
Requirement already satisfied: nest-asyncio in /Users/taanhtuan/miniconda
3/envs/churn/lib/python3.12/site-packages (from ipykernel==6.29.5->-r requ
irements.txt (line 10)) (1.6.0)
Requirement already satisfied: psutil in /Users/taanhtuan/miniconda3/envs/
churn/lib/python3.12/site-packages (from ipykernel==6.29.5->-r requirement
s.txt (line 10)) (7.1.0)
```

```
Requirement already satisfied: pyzmq>=24 in /Users/taanhtuan/miniconda3/en
vs/churn/lib/python3.12/site-packages (from ipykernel==6.29.5->-r requirem
ents.txt (line 10)) (27.1.0)
Requirement already satisfied: traitlets>=5.4.0 in /Users/taanhtuan/minico
nda3/envs/churn/lib/python3.12/site-packages (from ipykernel==6.29.5->-r r
equirements.txt (line 10)) (5.14.3)
Requirement already satisfied: widgetsnbextension~=4.0.10 in /Users/taanht
uan/miniconda3/envs/churn/lib/python3.12/site-packages (from ipywidgets==
8.1.2->-r requirements.txt (line 11)) (4.0.14)
Requirement already satisfied: jupyterlab-widgets~=3.0.10 in /Users/taanht
uan/miniconda3/envs/churn/lib/python3.12/site-packages (from ipywidgets==
8.1.2->-r requirements.txt (line 11)) (3.0.15)
Requirement already satisfied: anyio>=3.1.0 in /Users/taanhtuan/miniconda
3/envs/churn/lib/python3.12/site-packages (from jupyter-server<3,>=2.4.0->
notebook==7.2.2->-r requirements.txt (line 9)) (4.10.0)
Requirement already satisfied: argon2-cffi>=21.1 in /Users/taanhtuan/minic
onda3/envs/churn/lib/python3.12/site-packages (from jupyter-server<3,>=2.
4.0->notebook==7.2.2->-r requirements.txt (line 9)) (25.1.0)
Requirement already satisfied: jinja2>=3.0.3 in /Users/taanhtuan/miniconda
3/envs/churn/lib/python3.12/site-packages (from jupyter-server<3,>=2.4.0->
notebook==7.2.2->-r requirements.txt (line 9)) (3.1.6)
Requirement already satisfied: jupyter-events>=0.11.0 in /Users/taanhtuan/
miniconda3/envs/churn/lib/python3.12/site-packages (from jupyter-server<3,
>=2.4.0->notebook==7.2.2->-r requirements.txt (line 9)) (0.12.0)
Requirement already satisfied: jupyter-server-terminals>=0.4.4 in /Users/t
aanhtuan/miniconda3/envs/churn/lib/python3.12/site-packages (from jupyter-
server<3,>=2.4.0->notebook==7.2.2->-r requirements.txt (line 9)) (0.5.3)
Requirement already satisfied: nbconvert>=6.4.4 in /Users/taanhtuan/minico
nda3/envs/churn/lib/python3.12/site-packages (from jupyter-server<3,>=2.4.
0->notebook==7.2.2->-r requirements.txt (line 9)) (7.16.6)
Requirement already satisfied: nbformat>=5.3.0 in /Users/taanhtuan/minicon
da3/envs/churn/lib/python3.12/site-packages (from jupyter-server<3,>=2.4.0
->notebook==7.2.2->-r requirements.txt (line 9)) (5.10.4)
Requirement already satisfied: prometheus-client>=0.9 in /Users/taanhtuan/
miniconda3/envs/churn/lib/python3.12/site-packages (from jupyter-server<3,
>=2.4.0->notebook==7.2.2->-r requirements.txt (line 9)) (0.23.1)
Requirement already satisfied: send2trash>=1.8.2 in /Users/taanhtuan/minic
onda3/envs/churn/lib/python3.12/site-packages (from jupyter-server<3,>=2.
4.0->notebook==7.2.2->-r requirements.txt (line 9)) (1.8.3)
Requirement already satisfied: terminado>=0.8.3 in /Users/taanhtuan/minico
nda3/envs/churn/lib/python3.12/site-packages (from jupyter-server<3,>=2.4.
0->notebook==7.2.2->-r requirements.txt (line 9)) (0.18.1)
Requirement already satisfied: websocket-client>=1.7 in /Users/taanhtuan/m
iniconda3/envs/churn/lib/python3.12/site-packages (from jupyter-server<3,>
=2.4.0->notebook==7.2.2->-r requirements.txt (line 9)) (1.8.0)
Requirement already satisfied: async-lru>=1.0.0 in /Users/taanhtuan/minico
nda3/envs/churn/lib/python3.12/site-packages (from jupyterlab<4.3,>=4.2.0-
>notebook==7.2.2->-r requirements.txt (line 9)) (2.0.5)
Requirement already satisfied: httpx>=0.25.0 in /Users/taanhtuan/miniconda
3/envs/churn/lib/python3.12/site-packages (from jupyterlab<4.3,>=4.2.0->no
tebook==7.2.2->-r requirements.txt (line 9)) (0.28.1)
Requirement already satisfied: jupyter-lsp>=2.0.0 in /Users/taanhtuan/mini
conda3/envs/churn/lib/python3.12/site-packages (from jupyterlab<4.3,>=4.2.
0->notebook==7.2.2->-r requirements.txt (line 9)) (2.3.0)
Requirement already satisfied: setuptools>=40.8.0 in /Users/taanhtuan/mini
conda3/envs/churn/lib/python3.12/site-packages (from jupyterlab<4.3,>=4.2.
0->notebook==7.2.2->-r requirements.txt (line 9)) (78.1.1)
Requirement already satisfied: babel>=2.10 in /Users/taanhtuan/miniconda3/
envs/churn/lib/python3.12/site-packages (from jupyterlab-server<3,>=2.27.1
->notebook==7.2.2->-r requirements.txt (line 9)) (2.17.0)
```

```
Requirement already satisfied: json5>=0.9.0 in /Users/taanhtuan/miniconda
3/envs/churn/lib/python3.12/site-packages (from jupyterlab-server<3,>=2.2
7.1->notebook==7.2.2->-r requirements.txt (line 9)) (0.12.1)
Requirement already satisfied: jsonschema>=4.18.0 in /Users/taanhtuan/mini
conda3/envs/churn/lib/python3.12/site-packages (from jupyterlab-server<3,>
=2.27.1->notebook==7.2.2->-r requirements.txt (line 9)) (4.25.1)
Requirement already satisfied: requests>=2.31 in /Users/taanhtuan/minicond
a3/envs/churn/lib/python3.12/site-packages (from jupyterlab-server<3,>=2.2
7.1->notebook==7.2.2->-r requirements.txt (line 9)) (2.32.5)
Requirement already satisfied: idna>=2.8 in /Users/taanhtuan/miniconda3/en
vs/churn/lib/python3.12/site-packages (from anyio>=3.1.0->jupyter-server<
3,>=2.4.0->notebook==7.2.2->-r requirements.txt (line 9)) (3.10)
Requirement already satisfied: sniffio>=1.1 in /Users/taanhtuan/miniconda
3/envs/churn/lib/python3.12/site-packages (from anyio>=3.1.0->jupyter-serv
er<3,>=2.4.0->notebook==7.2.2->-r requirements.txt (line 9)) (1.3.1)
Requirement already satisfied: typing_extensions>=4.5 in /Users/taanhtuan/
miniconda3/envs/churn/lib/python3.12/site-packages (from anyio>=3.1.0->jup
yter-server<3,>=2.4.0->notebook==7.2.2->-r requirements.txt (line 9)) (4.1
5.0)
Requirement already satisfied: argon2-cffi-bindings in /Users/taanhtuan/mi
niconda3/envs/churn/lib/python3.12/site-packages (from argon2-cffi>=21.1->
jupyter-server<3,>=2.4.0->notebook==7.2.2->-r requirements.txt (line 9))
(25.1.0)
Requirement already satisfied: certifi in /Users/taanhtuan/miniconda3/env
s/churn/lib/python3.12/site-packages (from httpx>=0.25.0->jupyterlab<4.3,>
=4.2.0->notebook==7.2.2->-r requirements.txt (line 9)) (2025.8.3)
Requirement already satisfied: httpcore==1.* in /Users/taanhtuan/miniconda
3/envs/churn/lib/python3.12/site-packages (from httpx>=0.25.0->jupyterlab<
4.3,>=4.2.0->notebook==7.2.2->-r requirements.txt (line 9)) (1.0.9)
Requirement already satisfied: h11>=0.16 in /Users/taanhtuan/miniconda3/en
vs/churn/lib/python3.12/site-packages (from httpcore==1.*->httpx>=0.25.0->
jupyterlab<4.3,>=4.2.0->notebook==7.2.2->-r requirements.txt (line 9)) (0.
16.0)
Requirement already satisfied: decorator in /Users/taanhtuan/miniconda3/en
vs/churn/lib/python3.12/site-packages (from ipython>=7.23.1->ipykernel==6.
29.5->-r requirements.txt (line 10)) (5.2.1)
Requirement already satisfied: ipython-pygments-lexers in /Users/taanhtua
n/miniconda3/envs/churn/lib/python3.12/site-packages (from ipython>=7.23.1
->ipykernel==6.29.5->-r requirements.txt (line 10)) (1.1.1)
Requirement already satisfied: jedi>=0.16 in /Users/taanhtuan/miniconda3/e
nvs/churn/lib/python3.12/site-packages (from ipython>=7.23.1->ipykernel==
6.29.5->-r requirements.txt (line 10)) (0.19.2)
Requirement already satisfied: pexpect>4.3 in /Users/taanhtuan/miniconda3/
envs/churn/lib/python3.12/site-packages (from ipython>=7.23.1->ipykernel==
6.29.5->-r requirements.txt (line 10)) (4.9.0)
Requirement already satisfied: prompt_toolkit<3.1.0,>=3.0.41 in /Users/taa
nhtuan/miniconda3/envs/churn/lib/python3.12/site-packages (from ipython>=
7.23.1->ipykernel==6.29.5->-r requirements.txt (line 10)) (3.0.52)
Requirement already satisfied: pygments>=2.4.0 in /Users/taanhtuan/minicon
da3/envs/churn/lib/python3.12/site-packages (from ipython>=7.23.1->ipykern
el==6.29.5->-r requirements.txt (line 10)) (2.19.2)
Requirement already satisfied: stack_data in /Users/taanhtuan/miniconda3/e
nvs/churn/lib/python3.12/site-packages (from ipython>=7.23.1->ipykernel==
6.29.5->-r requirements.txt (line 10)) (0.6.3)
Requirement already satisfied: wcwidth in /Users/taanhtuan/miniconda3/env
s/churn/lib/python3.12/site-packages (from prompt_toolkit<3.1.0,>=3.0.41->
ipython>=7.23.1->ipykernel==6.29.5->-r requirements.txt (line 10)) (0.2.1
3)
Requirement already satisfied: parso<0.9.0,>=0.8.4 in /Users/taanhtuan/min
iconda3/envs/churn/lib/python3.12/site-packages (from jedi>=0.16->ipython>
=7.23.1->ipykernel==6.29.5->-r requirements.txt (line 10)) (0.8.5)
Requirement already satisfied: MarkupSafe>=2.0 in /Users/taanhtuan/minicon
da3/envs/churn/lib/python3.12/site-packages (from jinja2>=3.0.3->jupyter-s
erver<3,>=2.4.0->notebook==7.2.2->-r requirements.txt (line 9)) (3.0.2)
Requirement already satisfied: attrs>=22.2.0 in /Users/taanhtuan/miniconda
3/envs/churn/lib/python3.12/site-packages (from jsonschema>=4.18.0->jupyte
rlab-server<3,>=2.27.1->notebook==7.2.2->-r requirements.txt (line 9)) (2
5.3.0)
Requirement already satisfied: jsonschema-specifications>=2023.03.6 in /Us
ers/taanhtuan/miniconda3/envs/churn/lib/python3.12/site-packages (from jso
nschema>=4.18.0->jupyterlab-server<3,>=2.27.1->notebook==7.2.2->-r require
ments.txt (line 9)) (2025.9.1)
Requirement already satisfied: referencing>=0.28.4 in /Users/taanhtuan/min
iconda3/envs/churn/lib/python3.12/site-packages (from jsonschema>=4.18.0->
jupyterlab-server<3,>=2.27.1->notebook==7.2.2->-r requirements.txt (line
9)) (0.36.2)
Requirement already satisfied: rpds-py>=0.7.1 in /Users/taanhtuan/minicond
a3/envs/churn/lib/python3.12/site-packages (from jsonschema>=4.18.0->jupyt
erlab-server<3,>=2.27.1->notebook==7.2.2->-r requirements.txt (line 9))
(0.27.1)
Requirement already satisfied: platformdirs>=2.5 in /Users/taanhtuan/minic
onda3/envs/churn/lib/python3.12/site-packages (from jupyter-core!=5.0.*,>=
4.12->ipykernel==6.29.5->-r requirements.txt (line 10)) (4.4.0)
Requirement already satisfied: python-json-logger>=2.0.4 in /Users/taanhtu
an/miniconda3/envs/churn/lib/python3.12/site-packages (from jupyter-events
>=0.11.0->jupyter-server<3,>=2.4.0->notebook==7.2.2->-r requirements.txt
(line 9)) (3.3.0)
Requirement already satisfied: pyyaml>=5.3 in /Users/taanhtuan/miniconda3/
envs/churn/lib/python3.12/site-packages (from jupyter-events>=0.11.0->jupy
ter-server<3,>=2.4.0->notebook==7.2.2->-r requirements.txt (line 9)) (6.0.
2)
Requirement already satisfied: rfc3339-validator in /Users/taanhtuan/minic
onda3/envs/churn/lib/python3.12/site-packages (from jupyter-events>=0.11.0
->jupyter-server<3,>=2.4.0->notebook==7.2.2->-r requirements.txt (line 9))
(0.1.4)
Requirement already satisfied: rfc3986-validator>=0.1.1 in /Users/taanhtua
n/miniconda3/envs/churn/lib/python3.12/site-packages (from jupyter-events>
=0.11.0->jupyter-server<3,>=2.4.0->notebook==7.2.2->-r requirements.txt (l
ine 9)) (0.1.1)
Requirement already satisfied: fqdn in /Users/taanhtuan/miniconda3/envs/ch
urn/lib/python3.12/site-packages (from jsonschema[format-nongpl]>=4.18.0->
jupyter-events>=0.11.0->jupyter-server<3,>=2.4.0->notebook==7.2.2->-r requ
irements.txt (line 9)) (1.5.1)
Requirement already satisfied: isoduration in /Users/taanhtuan/miniconda3/
envs/churn/lib/python3.12/site-packages (from jsonschema[format-nongpl]>=
4.18.0->jupyter-events>=0.11.0->jupyter-server<3,>=2.4.0->notebook==7.2.2-
>-r requirements.txt (line 9)) (20.11.0)
Requirement already satisfied: jsonpointer>1.13 in /Users/taanhtuan/minico
nda3/envs/churn/lib/python3.12/site-packages (from jsonschema[format-nongp
l]>=4.18.0->jupyter-events>=0.11.0->jupyter-server<3,>=2.4.0->notebook==7.
2.2->-r requirements.txt (line 9)) (3.0.0)
Requirement already satisfied: rfc3987-syntax>=1.1.0 in /Users/taanhtuan/m
iniconda3/envs/churn/lib/python3.12/site-packages (from jsonschema[format-
nongpl]>=4.18.0->jupyter-events>=0.11.0->jupyter-server<3,>=2.4.0->noteboo
k==7.2.2->-r requirements.txt (line 9)) (1.1.0)
Requirement already satisfied: uri-template in /Users/taanhtuan/miniconda
3/envs/churn/lib/python3.12/site-packages (from jsonschema[format-nongpl]>
=4.18.0->jupyter-events>=0.11.0->jupyter-server<3,>=2.4.0->notebook==7.2.2
->-r requirements.txt (line 9)) (1.3.0)
Requirement already satisfied: webcolors>=24.6.0 in /Users/taanhtuan/minic
```

```
onda3/envs/churn/lib/python3.12/site-packages (from jsonschema[format-nong
pl]>=4.18.0->jupyter-events>=0.11.0->jupyter-server<3,>=2.4.0->notebook==
7.2.2->-r requirements.txt (line 9)) (24.11.1)
Requirement already satisfied: beautifulsoup4 in /Users/taanhtuan/minicond
a3/envs/churn/lib/python3.12/site-packages (from nbconvert>=6.4.4->jupyter
-server<3,>=2.4.0->notebook==7.2.2->-r requirements.txt (line 9)) (4.13.5)
Requirement already satisfied: bleach!=5.0.0 in /Users/taanhtuan/miniconda
3/envs/churn/lib/python3.12/site-packages (from bleach[css]!=5.0.0->nbconv
ert>=6.4.4->jupyter-server<3,>=2.4.0->notebook==7.2.2->-r requirements.txt
(line 9)) (6.2.0)
Requirement already satisfied: defusedxml in /Users/taanhtuan/miniconda3/e
nvs/churn/lib/python3.12/site-packages (from nbconvert>=6.4.4->jupyter-ser
ver<3,>=2.4.0->notebook==7.2.2->-r requirements.txt (line 9)) (0.7.1)
Requirement already satisfied: jupyterlab-pygments in /Users/taanhtuan/min
iconda3/envs/churn/lib/python3.12/site-packages (from nbconvert>=6.4.4->ju
pyter-server<3,>=2.4.0->notebook==7.2.2->-r requirements.txt (line 9)) (0.
3.0)
Requirement already satisfied: mistune<4,>=2.0.3 in /Users/taanhtuan/minic
onda3/envs/churn/lib/python3.12/site-packages (from nbconvert>=6.4.4->jupy
ter-server<3,>=2.4.0->notebook==7.2.2->-r requirements.txt (line 9)) (3.1.
4)
Requirement already satisfied: nbclient>=0.5.0 in /Users/taanhtuan/minicon
da3/envs/churn/lib/python3.12/site-packages (from nbconvert>=6.4.4->jupyte
r-server<3,>=2.4.0->notebook==7.2.2->-r requirements.txt (line 9)) (0.10.
2)
Requirement already satisfied: pandocfilters>=1.4.1 in /Users/taanhtuan/mi
niconda3/envs/churn/lib/python3.12/site-packages (from nbconvert>=6.4.4->j
upyter-server<3,>=2.4.0->notebook==7.2.2->-r requirements.txt (line 9))
(1.5.1)
Requirement already satisfied: webencodings in /Users/taanhtuan/miniconda
3/envs/churn/lib/python3.12/site-packages (from bleach!=5.0.0->bleach[cs
s]!=5.0.0->nbconvert>=6.4.4->jupyter-server<3,>=2.4.0->notebook==7.2.2->-r
requirements.txt (line 9)) (0.5.1)
Requirement already satisfied: tinycss2<1.5,>=1.1.0 in /Users/taanhtuan/mi
niconda3/envs/churn/lib/python3.12/site-packages (from bleach[css]!=5.0.0-
>nbconvert>=6.4.4->jupyter-server<3,>=2.4.0->notebook==7.2.2->-r requireme
nts.txt (line 9)) (1.4.0)
Requirement already satisfied: fastjsonschema>=2.15 in /Users/taanhtuan/mi
niconda3/envs/churn/lib/python3.12/site-packages (from nbformat>=5.3.0->ju
pyter-server<3,>=2.4.0->notebook==7.2.2->-r requirements.txt (line 9)) (2.
21.2)
Requirement already satisfied: ptyprocess>=0.5 in /Users/taanhtuan/minicon
da3/envs/churn/lib/python3.12/site-packages (from pexpect>4.3->ipython>=7.
23.1->ipykernel==6.29.5->-r requirements.txt (line 10)) (0.7.0)
Requirement already satisfied: six>=1.5 in /Users/taanhtuan/miniconda3/env
s/churn/lib/python3.12/site-packages (from python-dateutil>=2.8.2->pandas=
=2.2.2->-r requirements.txt (line 2)) (1.17.0)
Requirement already satisfied: charset_normalizer<4,>=2 in /Users/taanhtua
n/miniconda3/envs/churn/lib/python3.12/site-packages (from requests>=2.31-
>jupyterlab-server<3,>=2.27.1->notebook==7.2.2->-r requirements.txt (line
9)) (3.4.3)
Requirement already satisfied: urllib3<3,>=1.21.1 in /Users/taanhtuan/mini
conda3/envs/churn/lib/python3.12/site-packages (from requests>=2.31->jupyt
erlab-server<3,>=2.27.1->notebook==7.2.2->-r requirements.txt (line 9))
(2.5.0)
Requirement already satisfied: lark>=1.2.2 in /Users/taanhtuan/miniconda3/
envs/churn/lib/python3.12/site-packages (from rfc3987-syntax>=1.1.0->jsons
chema[format-nongpl]>=4.18.0->jupyter-events>=0.11.0->jupyter-server<3,>=
2.4.0->notebook==7.2.2->-r requirements.txt (line 9)) (1.2.2)
Requirement already satisfied: cffi>=1.0.1 in /Users/taanhtuan/miniconda3/
```

```
envs/churn/lib/python3.12/site-packages (from argon2-cffi-bindings->argon2
-cffi>=21.1->jupyter-server<3,>=2.4.0->notebook==7.2.2->-r requirements.tx
t (line 9)) (2.0.0)
Requirement already satisfied: pycparser in /Users/taanhtuan/miniconda3/en
vs/churn/lib/python3.12/site-packages (from cffi>=1.0.1->argon2-cffi-bindi
ngs->argon2-cffi>=21.1->jupyter-server<3,>=2.4.0->notebook==7.2.2->-r requ
irements.txt (line 9)) (2.23)
Requirement already satisfied: soupsieve>1.2 in /Users/taanhtuan/miniconda
3/envs/churn/lib/python3.12/site-packages (from beautifulsoup4->nbconvert>
=6.4.4->jupyter-server<3,>=2.4.0->notebook==7.2.2->-r requirements.txt (li
ne 9)) (2.8)
Requirement already satisfied: arrow>=0.15.0 in /Users/taanhtuan/miniconda
3/envs/churn/lib/python3.12/site-packages (from isoduration->jsonschema[fo
rmat-nongpl]>=4.18.0->jupyter-events>=0.11.0->jupyter-server<3,>=2.4.0->no
tebook==7.2.2->-r requirements.txt (line 9)) (1.3.0)
Requirement already satisfied: types-python-dateutil>=2.8.10 in /Users/taa
nhtuan/miniconda3/envs/churn/lib/python3.12/site-packages (from arrow>=0.1
5.0->isoduration->jsonschema[format-nongpl]>=4.18.0->jupyter-events>=0.11.
0->jupyter-server<3,>=2.4.0->notebook==7.2.2->-r requirements.txt (line
9)) (2.9.0.20250822)
Requirement already satisfied: llvmlite<0.46,>=0.45.0dev0 in /Users/taanht
uan/miniconda3/envs/churn/lib/python3.12/site-packages (from numba->shap==
0.45.1->-r requirements.txt (line 7)) (0.45.0)
Requirement already satisfied: executing>=1.2.0 in /Users/taanhtuan/minico
nda3/envs/churn/lib/python3.12/site-packages (from stack_data->ipython>=7.
23.1->ipykernel==6.29.5->-r requirements.txt (line 10)) (2.2.1)
Requirement already satisfied: asttokens>=2.1.0 in /Users/taanhtuan/minico
nda3/envs/churn/lib/python3.12/site-packages (from stack_data->ipython>=7.
23.1->ipykernel==6.29.5->-r requirements.txt (line 10)) (3.0.0)
Requirement already satisfied: pure-eval in /Users/taanhtuan/miniconda3/en
vs/churn/lib/python3.12/site-packages (from stack_data->ipython>=7.23.1->i
pykernel==6.29.5->-r requirements.txt (line 10)) (0.2.3)
```

2. Import Libraries

In [3]:
```python
import pandas as pd
import seaborn as sns
import numpy as np
import matplotlib.pyplot as plt
from matplotlib.ticker import PercentFormatter

from sklearn.model_selection import train_test_split
from sklearn.preprocessing import OneHotEncoder
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import Pipeline
from sklearn.ensemble import RandomForestClassifier
import shap

from sklearn.preprocessing import StandardScaler, OneHotEncoder
from sklearn.compose import ColumnTransformer
from sklearn.cluster import KMeans
from sklearn.metrics import silhouette_score
from sklearn.decomposition import PCA
```

3. Load Data & Quick Look

In [4]:
```
DATA_PATH = r"Dataset/saas_customer_churn.csv"
```

In [5]:
```
df = pd.read_csv(DATA_PATH)
```

```
/var/folders/6x/h489kd6s4lxbdnxrhw8mvt140000gn/T/ipykernel_44140/214458244
6.py:1: DtypeWarning: Columns (13) have mixed types. Specify dtype option
on import or set low_memory=False.
  df = pd.read_csv(DATA_PATH)
```

In [6]:
```
df.columns
```

Out[6]:
```
Index(['age', 'gender', 'security_no', 'region_category',
       'membership_category', 'joining_date', 'joined_through_referral',
       'referral_id', 'preferred_offer_types', 'medium_of_operation',
       'internet_option', 'last_visit_time', 'days_since_last_login',
       'avg_time_spent', 'avg_transaction_value', 'avg_frequency_login_d
ays',
       'points_in_wallet', 'used_special_discount',
       'offer_application_preference', 'past_complaint', 'complaint_stat
us',
       'feedback', 'churn_risk_score'],
      dtype='object')
```

In [7]:
```
df.shape
```

Out[7]:
```
(36992, 23)
```

In [8]:
```
df.head(5)
```

Out[8]:

	age	gender	security_no	region_category	membership_category	joining_date
0	18	F	XW0DQ7H	Village	Platinum Membership	17-08-2017
1	32	F	5K0N3X1	City	Premium Membership	28-08-2017
2	44	F	1F2TCL3	Town	No Membership	11-11-2016
3	37	M	VJGJ33N	City	No Membership	29-10-2016
4	31	F	SVZXCWB	City	No Membership	12-09-2017

5 rows × 23 columns

In [9]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 36992 entries, 0 to 36991
Data columns (total 23 columns):
 #   Column                        Non-Null Count  Dtype  
---  ------                        --------------  -----  
 0   age                           36992 non-null  int64  
 1   gender                        36992 non-null  object 
 2   security_no                   36992 non-null  object 
 3   region_category               31564 non-null  object 
 4   membership_category           36992 non-null  object 
 5   joining_date                  36992 non-null  object 
 6   joined_through_referral       36992 non-null  object 
 7   referral_id                   36992 non-null  object 
 8   preferred_offer_types         36704 non-null  object 
 9   medium_of_operation           36992 non-null  object 
 10  internet_option               36992 non-null  object 
 11  last_visit_time               36992 non-null  object 
 12  days_since_last_login         36992 non-null  int64  
 13  avg_time_spent                36992 non-null  object 
 14  avg_transaction_value         36992 non-null  float64
 15  avg_frequency_login_days      36992 non-null  object 
 16  points_in_wallet              33549 non-null  float64
 17  used_special_discount         36992 non-null  object 
 18  offer_application_preference  36992 non-null  object 
 19  past_complaint                36992 non-null  object 
 20  complaint_status              36992 non-null  object 
 21  feedback                      36990 non-null  object 
 22  churn_risk_score              36992 non-null  int64  
dtypes: float64(2), int64(3), object(18)
memory usage: 6.5+ MB
```

4. Missing, Duplicates, Wrong Format Checking

Missing

In [10]:
```
missing = df.isna().sum().sort_values(ascending=False)
(pd.DataFrame({"missing": missing, "missing_%": (df.isna().mean()*100).ro
 .query("missing > 0")
 .head(20))
```

Out[10]:

	missing	missing_%
feedback	2	0.01
points_in_wallet	3443	9.31
preferred_offer_types	288	0.78
region_category	5428	14.67

Missing Values Evaluation

- `feedback` : negligible (0.01%), fill with `"Unknown"` .
- `points_in_wallet` : moderate (9.3%), impute with median.
- `preferred_offer_types` : low (0.8%), fill with mode/ `Unknown` .

- `region_category` : higher (14.7%), fill with `Unknown` .

Duplicates

In [11]:
```
df.duplicated().sum()
```

Out[11]:
```
0
```

Duplicate Check

- No duplicate rows found → dataset records are unique.

Numeric summary (pre-clean)

In [12]:
```
df.describe(include=[np.number])
```

Out[12]:

	age	days_since_last_login	avg_transaction_value	points_in_walle
count	36992.000000	36992.000000	36992.000000	33549.00000
mean	37.118161	-41.915576	29271.194003	686.88219
std	15.867412	228.819900	19444.806226	194.06362
min	10.000000	-999.000000	800.460000	-760.66123
25%	23.000000	8.000000	14177.540000	616.15000
50%	37.000000	12.000000	27554.485000	697.62000
75%	51.000000	16.000000	40855.110000	763.95000
max	64.000000	26.000000	99914.050000	2069.0697

Numeric Summary Evaluation

- `age` : Values 10–64, reasonable.
- `days_since_last_login` : Invalid negatives (min = -999), needs cleaning.
- `avg_transaction_value` : Wide range (800–99914), potential outliers.
- `points_in_wallet` : Invalid negatives (min = -760), needs correction.
- `churn_risk_score` : Bounded 0–1, distribution looks valid.

Categorical snapshot (pre-clean)

In [13]:
```
df.describe(include=["object"])
```

Out[13]:

	gender	security_no	region_category	membership_category	joining_date
count	36992	36992	31564	36992	36992
unique	3	36992	3	6	1100
top	F	XW0DQ7H	Town	Basic Membership	02-06-2015
freq	18490	1	14128	7724	55

In [14]:

```python
# unique values for categorical columns
cat_cols = df.select_dtypes(exclude=[np.number]).columns

for c in cat_cols:
    uniques = df[c].dropna().unique()
    print(f"\n=== {c} | {len(uniques)} unique values ===")
    print(uniques)
```

```
=== gender | 3 unique values ===
['F' 'M' 'Unknown']

=== security_no | 36992 unique values ===
['XW0DQ7H' '5K0N3X1' '1F2TCL3' ... 'XK1IM9H' 'K6VTP1Z' 'LBX0GLR']

=== region_category | 3 unique values ===
['Village' 'City' 'Town']

=== membership_category | 6 unique values ===
['Platinum Membership' 'Premium Membership' 'No Membership'
 'Gold Membership' 'Silver Membership' 'Basic Membership']

=== joining_date | 1100 unique values ===
['17-08-2017' '28-08-2017' '11-11-2016' ... '11-12-2017' '25-09-2016'
 '15-04-2017']

=== joined_through_referral | 3 unique values ===
['No' '?' 'Yes']

=== referral_id | 11359 unique values ===
['xxxxxxxx' 'CID21329' 'CID12313' ... 'CID60808' 'CID10431' 'CID45477']

=== preferred_offer_types | 3 unique values ===
['Gift Vouchers/Coupons' 'Credit/Debit Card Offers' 'Without Offers']

=== medium_of_operation | 4 unique values ===
['?' 'Desktop' 'Smartphone' 'Both']

=== internet_option | 3 unique values ===
['Wi-Fi' 'Mobile_Data' 'Fiber_Optic']

=== last_visit_time | 30101 unique values ===
['16:08:02' '12:38:13' '22:53:21' ... '4:14:05' '9:50:03' '1:39:52']

=== avg_time_spent | 27525 unique values ===
['300.63' '306.34' '516.16' ... 154.94 482.61 79.18]

=== avg_frequency_login_days | 1654 unique values ===
['17' '10' '22' ... '-9.325511142' '-8.759329713' '27.83992744']

=== used_special_discount | 4 unique values ===
['Yes' 'No' 'yes' 'True']

=== offer_application_preference | 5 unique values ===
['Yes' 'No' 'yes' 'True' 'YES']

=== past_complaint | 4 unique values ===
['No' 'Yes' 'yes' 'YES']

=== complaint_status | 5 unique values ===
['Not Applicable' 'Solved' 'Solved in Follow-up' 'Unsolved'
 'No Information Available']

=== feedback | 12 unique values ===
['Products always in Stock' 'Quality Customer Care' 'Poor Website'
 'No reason specified' 'Poor Product Quality' 'Poor Customer Service'
 'Too many ads' 'User Friendly Website' 'Reasonable Price' 'alkfjlks'
 'XXXXXXX' 'Q']
```

Categorical Summary Evaluation

- `gender` : Clean (3 values: F, M, Unknown).
- `region_category` : 3 values, but ~15% missing → fill with `Unknown` .
- `membership_category` : 6 valid categories, no issues.
- `joined_through_referral` : Inconsistent (`Yes/No/?`) → normalize.
- `preferred_offer_types` : 3 values, some missing → fill `Unknown` .
- `medium_of_operation` : Contains `?` → replace with `Unknown` .
- `used_special_discount` , `offer_application_preference` , `past_complaint` : Mixed casing (`Yes/No/True`) → standardize.
- `complaint_status` : 5 categories, valid but can be grouped (`Solved` , `Unsolved` , `Not Applicable`).
- `feedback` : 12 categories, includes noise/typos (`XXXXXXX` , `Q`) → map to `Unknown/Other` .
- High-cardinality IDs (`security_no` , `referral_id`) → drop from analysis.
- Some columns like `avg_time_spent` , `avg_frequency_login_days` should be Numeric, but exist on Categorical → Some values is non-numeric → Need to check them more

Date Time Column - Format Checking

In [15]:
```python
# detect which date formats appear in `joining_date`
s = df["joining_date"].astype(str).str.strip()

fmts = [
    "%d-%m-%Y", "%d/%m/%Y", "%Y-%m-%d",
    "%m-%d-%Y", "%d-%b-%Y", "%b %d, %Y", "%d %b %Y"
]

seen = pd.Series(False, index=s.index)
results = []

for f in fmts:
    ok = pd.to_datetime(s, format=f, errors="coerce").notna() & ~seen
    if ok.any():
        results.append((f, int(ok.sum()), s[ok].head(3).tolist()))
        seen |= ok

# anything else
other = (~seen) & s.ne("") & s.ne("NaT")
if other.any():
    results.append(("unrecognized/other", int(other.sum()), s[other].head

# print summary
for f, cnt, ex in results:
    print(f"{f:>15}  -> {cnt:>6} rows   examples: {ex}")
```

```
       %d-%m-%Y  ->  36987 rows   examples: ['17-08-2017', '28-08-2017',
'11-11-2016']
       %d/%m/%Y  ->      3 rows   examples: ['12/07/2017', '17/08/2017',
'17/08/2017']
       %Y-%m-%d  ->      1 rows   examples: ['2017-03-06']
       %d-%b-%Y  ->      1 rows   examples: ['16-Jan-2017']
```

Date Time Column Evaluation

`joining_date`

- Mostly `dd-mm-yyyy` → consistent.
- Few outliers: `dd/mm/yyyy`, `yyyy-mm-dd`, `dd-Mon-yyyy`.
- Normalize all to ISO `YYYY-MM-DD` and audit those 5 rows.

avg_time_spent & avg_frequency_login_days

In [16]:
```python
# Cell — Check which "should-be-numeric" columns contain non-numeric valu
should_be_numeric = ["avg_time_spent", "avg_frequency_login_days"]

import re
pat_num = re.compile(r"^[+-]?(\d+(\.\d+)?|\.\d+)(e[+-]?\d+)?$")  # int/fl

for col in should_be_numeric:
    s = df[col].astype(str).str.strip()
    bad = s[(s != "") & (s.str.lower() != "nan") & (~s.map(lambda x: bool
    print(f"\n[{col}] non-numeric entries: {bad.shape[0]} rows")
    if not bad.empty:
        print("Top tokens:")
        print(bad.value_counts().head(10))
        print("Examples:", bad.unique()[:5].tolist())
```

```
[avg_time_spent] non-numeric entries: 2 rows
Top tokens:
avg_time_spent
$206.72        1
$1076.928992   1
Name: count, dtype: int64
Examples: ['$206.72', '$1076.928992']

[avg_frequency_login_days] non-numeric entries: 3522 rows
Top tokens:
avg_frequency_login_days
Error    3522
Name: count, dtype: int64
Examples: ['Error']
```

avg_time_spent & avg_frequency_login_days

- **avg_time_spent:** Mostly numeric, but 2 rows contain currency-formatted values (`$206.72`, `$1076.928992`) → need cleaning (`$` removal, convert to float).
- **avg_frequency_login_days:** 3,522 rows labeled as `'Error'` instead of numbers (~9.5% of data) → impute (e.g., median).

5. Data Cleaning

1. **Drop IDs:** `security_no` , `referral_id` .
2. **Normalize text:** strip whitespace; replace `?` , `XXXXXXX` , `Q` → `Unknown` ; standardize booleans (`Yes/No`); optionally group `complaint_status` .
3. **Unify dates:** parse all `joining_date` formats → ISO `YYYY-MM-DD` (audit 5 outliers).
4. **Fix numerics:**
   - `avg_time_spent` : remove `$` , convert to float.
   - `avg_frequency_login_days` : replace `'Error'` → **0** (assume no activity).
   - `days_since_last_login` : replace negatives → **median**.
   - `points_in_wallet` : replace negatives → **0** (no balance).
5. **Impute & cap:** fill `region_category` / `preferred_offer_types` / `feedback` → `Unknown` ; impute missing numerics with **mean/median**; cap extreme outliers at 99th percentile.
6. **Finalize:** drop duplicates (none found), set clean dtypes (categoricals/booleans), save cleaned dataset.

Drop IDs

```
In [17]: drop_cols = ["security_no", "referral_id"]
         df = df.drop(columns=[c for c in drop_cols if c in df.columns], errors="i
         df.shape
```

```
Out[17]: (36992, 21)
```

Normalize text columns

```
In [18]: for col in df.select_dtypes(include="object").columns:
             df[c] = (
                 df[col]
                 .astype(str)
                 .str.strip()                                # trim ends
                 .str.replace(r"\s+", " ", regex=True)       # collapse multiple sp
                 .replace({"?": "Unknown",                   # placeholders → 'Unkn
                           "XXXXXXX": "Unknown",
                           "Q": "Unknown"})
             )
```

Standardize booleans to "Yes"/"No"

```
In [19]: YES = {"yes", "y", "true", "1"}
         NO  = {"no", "n", "false", "0"}

         bool_cols = ["used_special_discount", "offer_application_preference", "pa
         for c in bool_cols:
```

```
    if c in df.columns:
        df[c] = df[c].map(lambda x: "Yes" if str(x).lower() in YES else (

if "joined_through_referral" in df.columns:
    df["joined_through_referral"] = df["joined_through_referral"].str.low
        {"yes": "Yes", "no": "No", "unknown": "Unknown"}
    ).fillna("Unknown")
```

Parse dates (joining_date → datetime), keep ISO string if desired

In [20]:
```
if "joining_date" in df.columns:
    jd = pd.to_datetime(df["joining_date"], errors="coerce", dayfirst=Tru
    # replace failures with the column's mode date (or earliest valid dat
    fill_date = jd.dropna().mode().iloc[0] if jd.notna().any() else pd.Ti
    jd = jd.fillna(fill_date)
    df["joining_date"] = jd
    df["joining_date_iso"] = df["joining_date"].dt.strftime("%Y-%m-%d")
```

```
/var/folders/6x/h489kd6s4lxbdnxrhw8mvt140000gn/T/ipykernel_44140/39952098
1.py:2: UserWarning: The argument 'infer_datetime_format' is deprecated an
d will be removed in a future version. A strict version of it is now the d
efault, see https://pandas.pydata.org/pdeps/0004-consistent-to-datetime-pa
rsing.html. You can safely remove this argument.
  jd = pd.to_datetime(df["joining_date"], errors="coerce", dayfirst=True,
infer_datetime_format=True)
```

Fix numerics

In [21]:
```
# avg_time_spent: remove $/commas, to float
s = df["avg_time_spent"].astype(str).str.replace(r"[\$,]", "", regex=True
df["avg_time_spent"] = pd.to_numeric(s, errors="coerce")
# Fill any remaining gaps with column median
df["avg_time_spent"] = df["avg_time_spent"].fillna(df["avg_time_spent"].m
```

In [22]:
```
# avg_frequency_login_days: 'Error' -> 0, then numeric
s = df["avg_frequency_login_days"].astype(str).str.strip()
s = s.mask(s.str.lower().eq("error"), "0")
df["avg_frequency_login_days"] = pd.to_numeric(s, errors="coerce").fillna
```

In [23]:
```
# days_since_last_login: negatives -> median of valid non-negative values
nonneg_median = df.loc[df["days_since_last_login"] >= 0, "days_since_last
df.loc[df["days_since_last_login"] < 0, "days_since_last_login"] = nonneg
```

In [24]:
```
# points_in_wallet: negatives -> 0 (no balance)
df["points_in_wallet"] = pd.to_numeric(df["points_in_wallet"], errors="co
df.loc[df["points_in_wallet"] < 0, "points_in_wallet"] = 0
df["points_in_wallet"] = df["points_in_wallet"].fillna(0)
```

In [25]:
```
### Impute categoricals & numerics (mean/median), then cap outliers
```

In [26]:
```
# Categorical: fill with "Unknown"
for c in ["region_category", "preferred_offer_types", "feedback", "medium
    if c in df.columns:
        df[c] = df[c].fillna("Unknown").replace({"": "Unknown"})
```

In [27]:
```
# Numeric: fill remaining missing with median (wallet by membership)
med_by_mem = df.groupby("membership_category")["points_in_wallet"].transf
df["points_in_wallet"] = df["points_in_wallet"].where(df["points_in_walle
df["points_in_wallet"] = df["points_in_wallet"].fillna(df["points_in_wall
```

In [28]:
```
# Numeric: fill remaining missing with median
num_cols = df.select_dtypes(include=[np.number]).columns.tolist()
for c in num_cols:
    if df[c].isna().any():
        # default: median for robustness
        df[c] = df[c].fillna(df[c].median())
```

In [29]:
```
# Outlier capping (99th percentile) for key behavior/revenue fields
for c in ["avg_transaction_value", "points_in_wallet", "avg_time_spent",
    if c in df.columns:
        hi = df[c].quantile(0.99)
        lo = 0 if c in ["points_in_wallet", "avg_frequency_login_days"] e
        df[c] = df[c].clip(lower=lo, upper=hi)
```

6. Final Quality Check

In [30]:
```
# Dtypes & missing after cleaning
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 36992 entries, 0 to 36991
Data columns (total 22 columns):
 #   Column                        Non-Null Count  Dtype
---  ------                        --------------  -----
 0   age                           36992 non-null  int64
 1   gender                        36992 non-null  object
 2   region_category               36992 non-null  object
 3   membership_category           36992 non-null  object
 4   joining_date                  36992 non-null  datetime64[ns]
 5   joined_through_referral       36992 non-null  object
 6   preferred_offer_types         36992 non-null  object
 7   medium_of_operation           36992 non-null  object
 8   internet_option               36992 non-null  object
 9   last_visit_time               36992 non-null  object
 10  days_since_last_login         36992 non-null  int64
 11  avg_time_spent                36992 non-null  float64
 12  avg_transaction_value         36992 non-null  float64
 13  avg_frequency_login_days      36992 non-null  float64
 14  points_in_wallet              36992 non-null  float64
 15  used_special_discount         36992 non-null  object
 16  offer_application_preference  36992 non-null  object
 17  past_complaint                36992 non-null  object
 18  complaint_status              36992 non-null  object
 19  feedback                      36992 non-null  object
 20  churn_risk_score              36992 non-null  int64
 21  joining_date_iso              36992 non-null  object
dtypes: datetime64[ns](1), float64(4), int64(3), object(14)
memory usage: 6.2+ MB
```

In [31]:
```
# Remaining missing
miss = df.isna().sum().sort_values(ascending=False)
```

```
pd.DataFrame({"missing": miss, "missing_%": (df.isna().mean()*100).round(
```

Out[31]:

	missing	missing_%
age	0	0.0
avg_frequency_login_days	0	0.0
avg_time_spent	0	0.0
avg_transaction_value	0	0.0
churn_risk_score	0	0.0
complaint_status	0	0.0
days_since_last_login	0	0.0
feedback	0	0.0
gender	0	0.0
internet_option	0	0.0
joined_through_referral	0	0.0
joining_date	0	0.0
joining_date_iso	0	0.0
last_visit_time	0	0.0
medium_of_operation	0	0.0

In [32]:
```
# Target distribution
df["churn_risk_score"].value_counts(dropna=False)
```

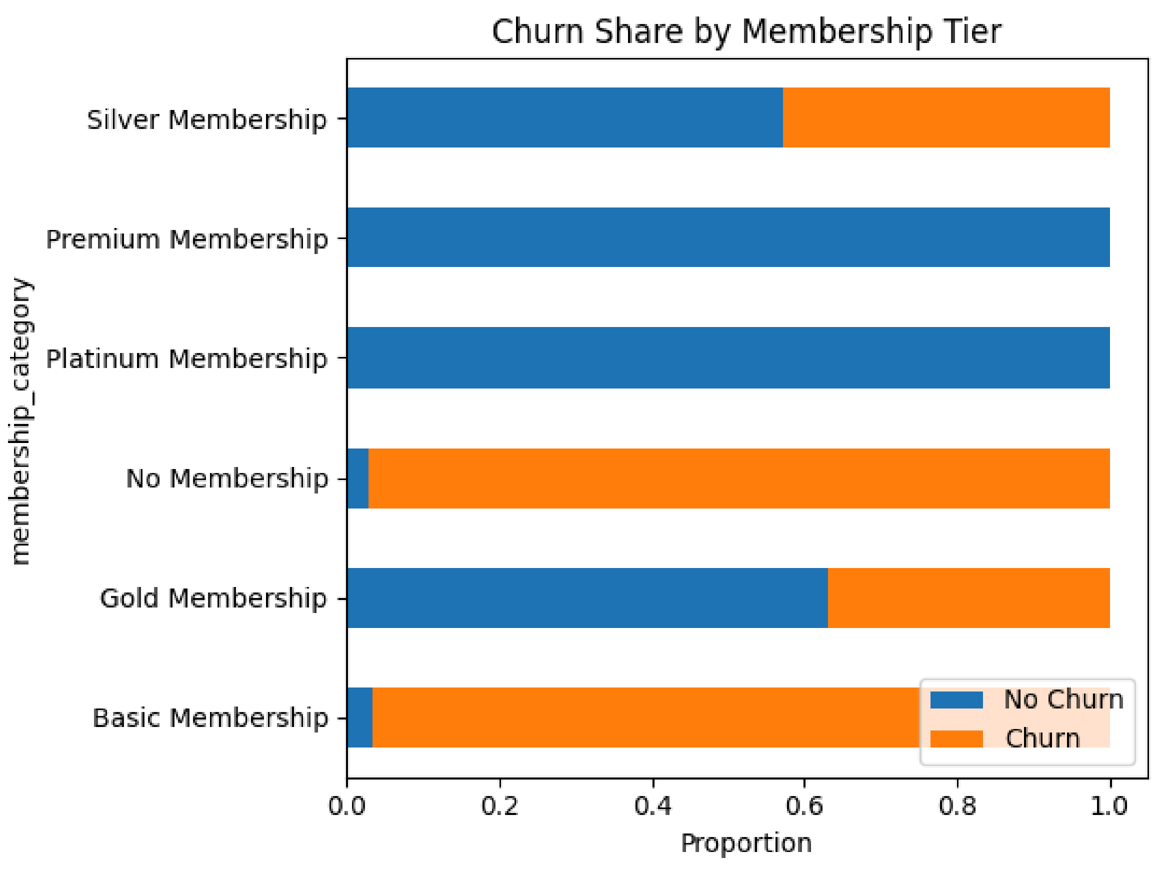
Out[32]:
```
churn_risk_score
1    20012
0    16980
Name: count, dtype: int64
```

7. Explanatory EDA — 10 Business Questions

Q1. Which membership tiers churn more?

In [33]:
```
ct = pd.crosstab(df["membership_category"], df["churn_risk_score"], norma
ct = ct.rename(columns={0.0:"No Churn", 1.0:"Churn"}).loc[ct.index]

ax = ct.plot(kind="barh", stacked=True)
ax.set_title("Churn Share by Membership Tier")
ax.set_xlabel("Proportion")
ax.legend(loc="lower right")
plt.tight_layout(); plt.show()
```



Churn by Membership — Conclusion

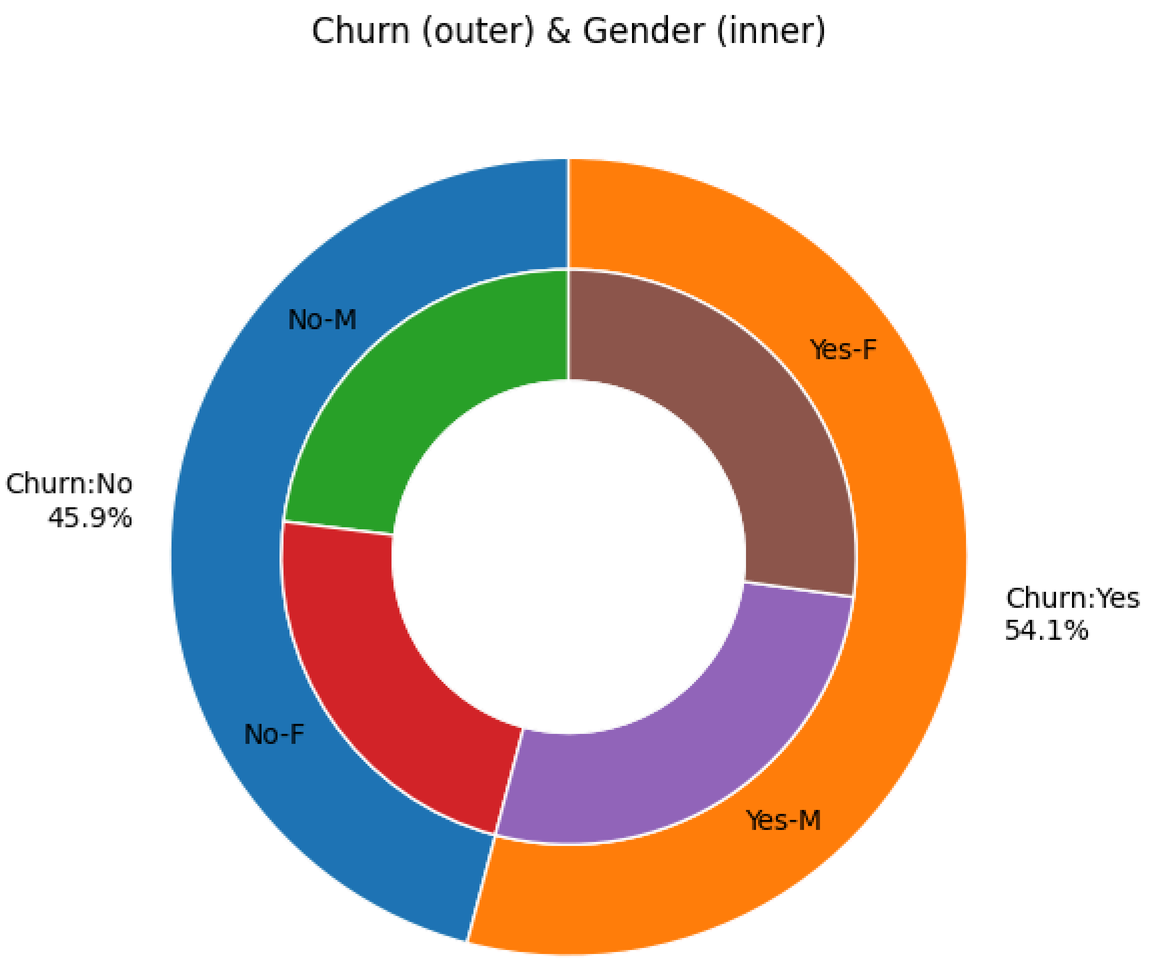
- **Highest churn:** No Membership, Basic → weak perceived value; target upgrades/onboarding.
- **Mid churn:** Silver, Gold → better but still meaningful; reinforce value.
- **Lowest churn:** Platinum, Premium → near-zero churn; maintain benefits.
- **Action:** Prioritize retention offers for entry tiers; protect premium experience.

Q2. Do churn rates differ by gender, and how is each churn group composed by gender?

In [34]:
```
# Simple nested donut: outer = churn share, inner = gender split within e
# Prep labels
df["ChurnLabel"] = df["churn_risk_score"].map({0:"No", 1:"Yes"})
df["Gender2"] = df["gender"].fillna("Unknown").map({"M":"M","F":"F"}).fil

# Data
outer = df["ChurnLabel"].value_counts().reindex(["No","Yes"])
inner_tbl = pd.crosstab(df["ChurnLabel"], df["Gender2"]).reindex(index=["
inner = inner_tbl.to_numpy().ravel()  # [No-M, No-F, Yes-M, Yes-F]

# Plot
fig, ax = plt.subplots(figsize=(6,6))
ax.pie(outer, labels=[f"Churn:{k}\n{v/outer.sum():.1%}" for k,v in outer.
       radius=1.0, wedgeprops=dict(width=0.28, edgecolor="white"), starta
ax.pie(inner, labels=["No-M","No-F","Yes-M","Yes-F"],
       radius=0.72, wedgeprops=dict(width=0.28, edgecolor="white"), start
ax.add_artist(plt.Circle((0,0), 0.44, color="white"))
ax.set_title("Churn (outer) & Gender (inner)")
plt.tight_layout(); plt.show()
```



In [35]:
```
df = df.drop('Gender2', axis=1)
```

Gender vs Churn — Conclusion

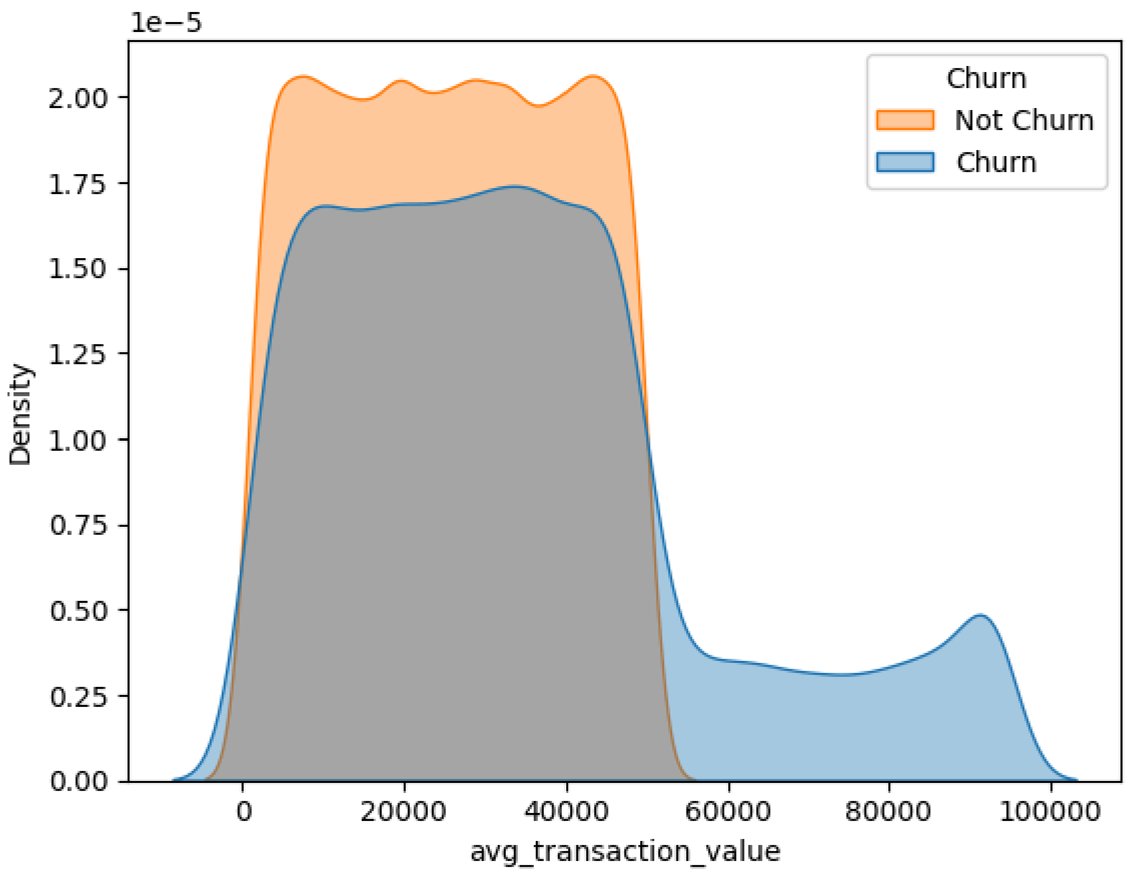
- Both **male and female customers churn significantly**, with no major gender gap.
- Gender alone does **not appear to be a strong differentiator of churn**.

Recommendation:
Focus churn analysis on **behavioral features** (usage frequency, spending, complaints) rather than gender. Gender can still be used for **marketing segmentation**, but not as a primary churn predictor.

Q3. Do customers with higher average transaction value churn more frequently?

In [36]:
```
sns.kdeplot(
    data=df.assign(ChurnLabel=df["churn_risk_score"].map({0:"Not Churn",
    x="avg_transaction_value", hue="ChurnLabel", fill=True, common_norm=F
)
```

```
plt.legend(title="Churn", labels=["Not Churn", "Churn"])
plt.show()
```



Avg Transaction Value vs Churn — Conclusion

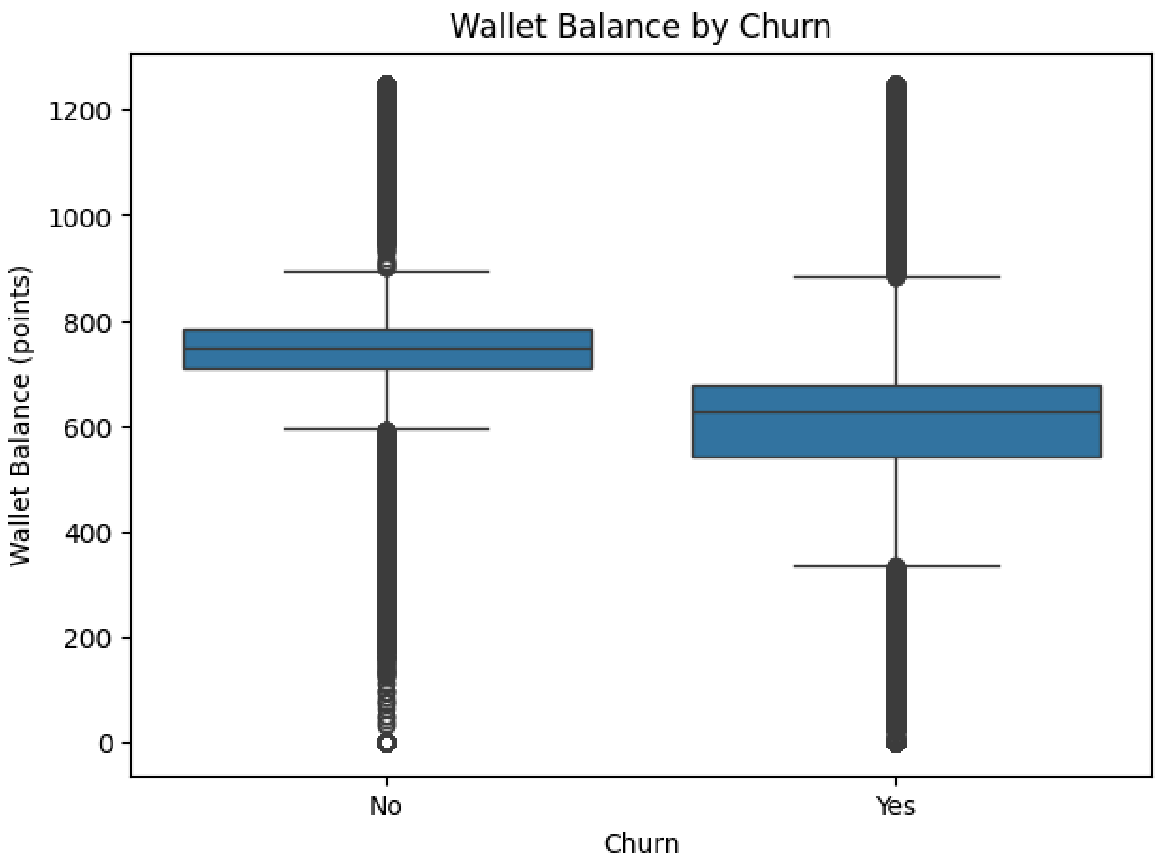
- Customers with **lower avg transaction values (0–40K)** are more often **not churn** → low spenders are more stable.
- Customers with **higher transaction values (60K–100K)** show higher **churn density** → big spenders are more at risk.
- Suggests **price sensitivity**: heavy spenders may churn if they feel costs outweigh benefits.

Recommendation:
Target high-transaction-value customers with **loyalty rewards, premium service, or discounts** to reduce churn.

Q4. Do customers with higher wallet balances churn less?

```
In [37]: df["ChurnLabel"] = df["churn_risk_score"].map({0:"No", 1:"Yes"})
         df["points_in_wallet"] = pd.to_numeric(df["points_in_wallet"], errors="co

         plt.figure(figsize=(7,5))
         sns.boxplot(data=df, x="ChurnLabel", y="points_in_wallet")
         plt.xlabel("Churn"); plt.ylabel("Wallet Balance (points)")
         plt.title("Wallet Balance by Churn")
         plt.show()
```



Box Plot Interpretation: Wallet Balance by Churn

Median Balance

- Non-churned customers (No) have a higher median wallet balance (~750).
- Churned customers (Yes) have a lower median (~620).
  => Suggests customers with more wallet balance are less likely to churn.

Spread (IQR – interquartile range)

- Non-churn group has a tighter spread (most balances between ~600–800).
- Churn group shows a wider spread, meaning more variation in balances.

Outliers

- Both groups have extreme values (very low and very high balances).
- Churned customers show more low-balance cases.

Overall Trend

- Higher wallet balances are linked to lower churn.
- Customers with low wallet balances appear more at risk of churn.

Q5. Is there a churn difference between customers with zero vs. non-zero wallet points?

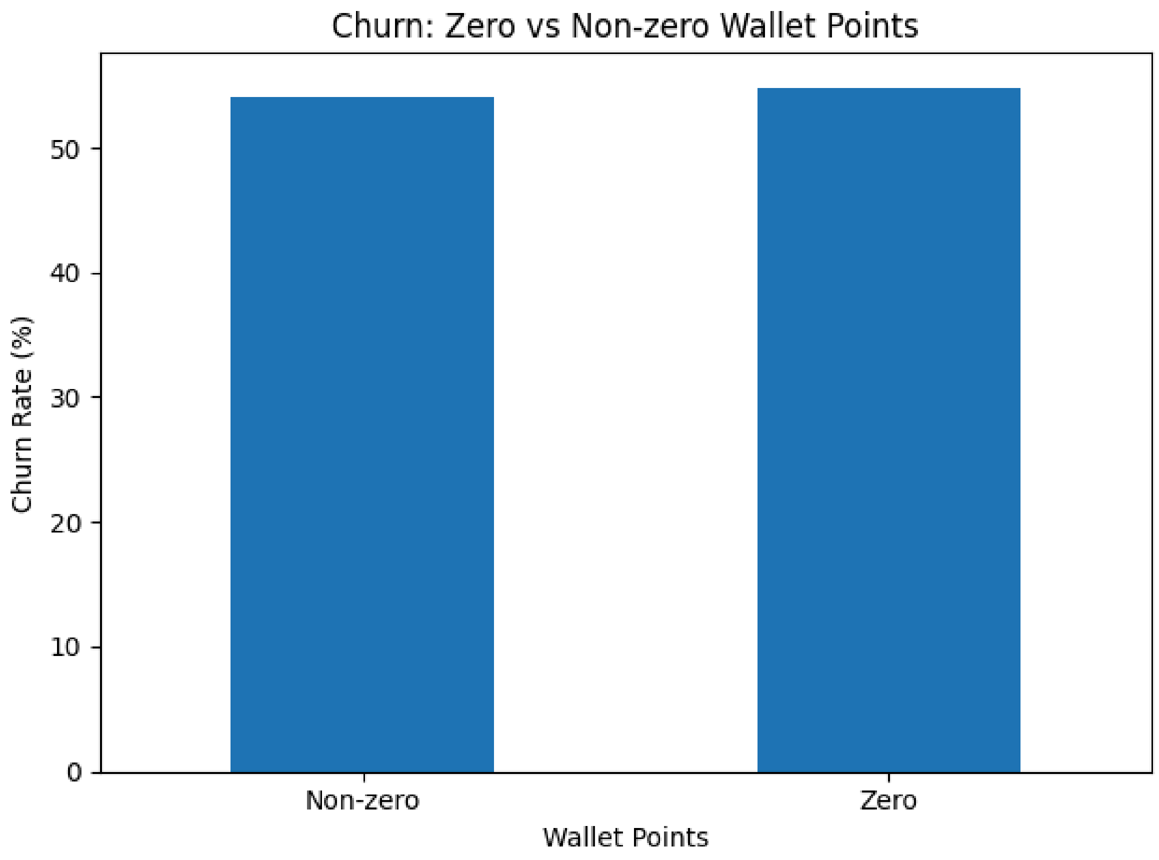
In [38]:
```
# make numeric (safeguard) and build flag
pts = pd.to_numeric(df["points_in_wallet"], errors="coerce").fillna(0)
```

```
flag = (pts <= 0).map({True: "Zero", False: "Non-zero"})

rate = df.groupby(flag)["churn_risk_score"].mean().mul(100).round(2)
print("Churn rate (%):\n", rate)

ax = rate.plot(kind="bar", title="Churn: Zero vs Non-zero Wallet Points")
ax.set_ylabel("Churn Rate (%)"); ax.set_xlabel("Wallet Points")
plt.xticks(rotation=0); plt.tight_layout(); plt.show()
```

```
Churn rate (%):
 points_in_wallet
Non-zero    54.02
Zero        54.85
Name: churn_risk_score, dtype: float64
```



Zero vs Non-zero Wallet Points — Conclusion

- Churn is ~54–55% for both groups (difference < 1%).
- Simply having wallet points does **not** meaningfully reduce churn.
- Focus retention on other drivers (usage, tier, offers).

Q6. Does higher login frequency correlate with lower churn risk?

In [39]:
```
# Build the feature as a Series (no mutation of df)
days = pd.to_numeric(df["avg_frequency_login_days"], errors="coerce")
login_freq_per_week = (7 / days.replace(0, np.nan)).clip(lower=0)
```

In [40]:
```
df_freq = pd.DataFrame({
    "login_freq_per_week": login_freq_per_week,
    "churn_risk_score": df["churn_risk_score"].astype(int)
}, index=df.index)
```

In [41]:
```
# Monotonic relationship (Spearman)
corr = df_freq[["login_freq_per_week", "churn_risk_score"]].corr(method="
print(f"Spearman corr = {corr:.3f}  (negative ⇒ higher frequency, lower c
```

```
Spearman corr = -0.151  (negative ⇒ higher frequency, lower churn)
```

In [42]:
```
# Quintile ranges (≈20% each)
freq = df_freq["login_freq_per_week"].dropna()
bins_intervals = pd.qcut(freq, 5, duplicates="drop")          # interva
intervals = bins_intervals.cat.categories
counts    = bins_intervals.value_counts(sort=False)

print("\nLogin frequency quintile ranges (logins/week):")
labels = ["Q1(low)", "Q2", "Q3", "Q4", "Q5(high)"]
for lbl, iv, n in zip(labels, intervals, counts):
    print(f"- {lbl}: {iv.left:.3f} → {iv.right:.3f}  (n={n}, {n/len(freq)
```

```
Login frequency quintile ranges (logins/week):
- Q1(low): 0.169 → 0.292  (n=7347, 22.4% of users)
- Q2: 0.292 → 0.389  (n=6995, 21.3% of users)
- Q3: 0.389 → 0.500  (n=5355, 16.3% of users)
- Q4: 0.500 → 0.778  (n=6623, 20.2% of users)
- Q5(high): 0.778 → 760.200  (n=6467, 19.7% of users)
```

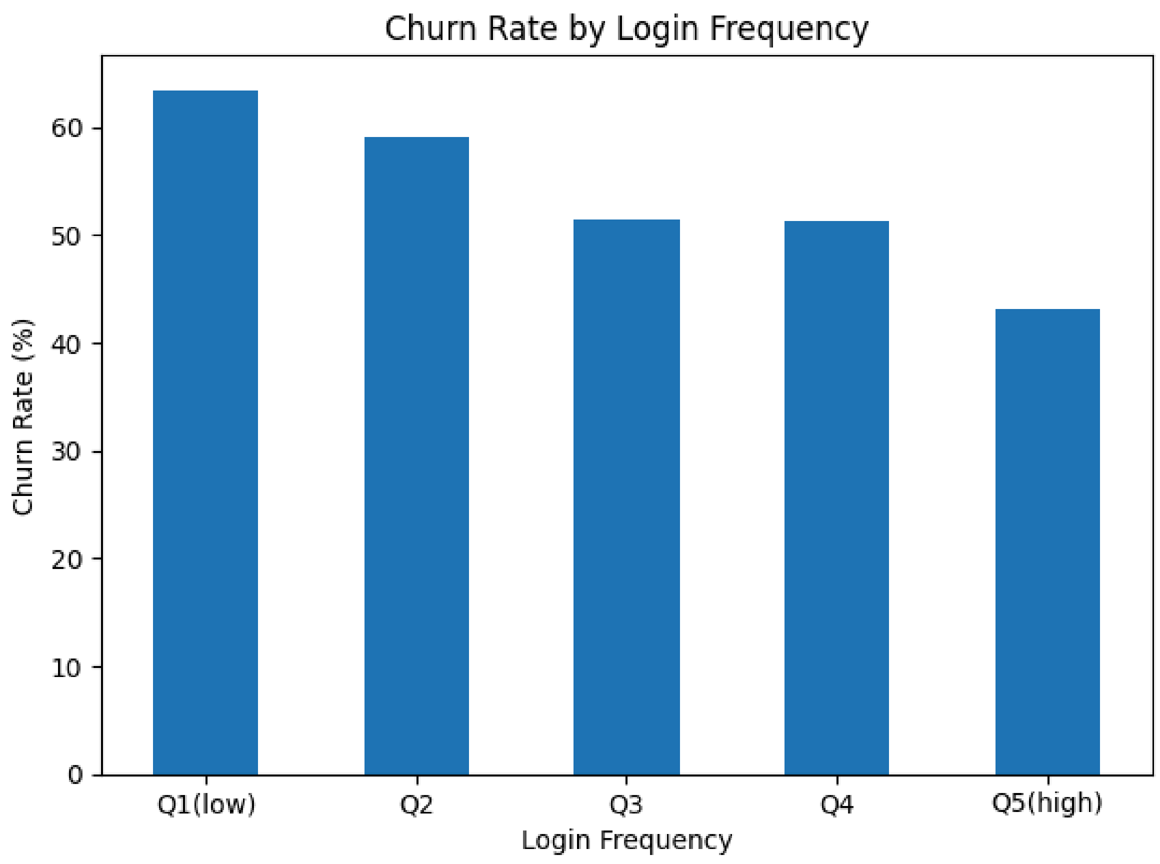
In [43]:
```
# Churn rate by quintile
bins = pd.qcut(freq, 5, labels=labels, duplicates="drop")
tmp  = df_freq.loc[freq.index].assign(freq_bin=bins)
rate = tmp.groupby("freq_bin")["churn_risk_score"].mean().mul(100).round(

print("\nChurn rate by login-frequency bucket (%):\n", rate)

ax = rate.plot(kind="bar", title="Churn Rate by Login Frequency")
ax.set_ylabel("Churn Rate (%)"); ax.set_xlabel("Login Frequency")
plt.xticks(rotation=0); plt.tight_layout(); plt.show()
```

```
Churn rate by login-frequency bucket (%):
 freq_bin
Q1(low)     63.5
Q2          59.1
Q3          51.4
Q4          51.3
Q5(high)    43.1
Name: churn_risk_score, dtype: float64
```

```
/var/folders/6x/h489kd6s4lxbdnxrhw8mvt140000gn/T/ipykernel_44140/131340147
6.py:4: FutureWarning: The default of observed=False is deprecated and wil
l be changed to True in a future version of pandas. Pass observed=False to
retain current behavior or observed=True to adopt the future default and s
ilence this warning.
  rate = tmp.groupby("freq_bin")["churn_risk_score"].mean().mul(100).round
(1)
```



Login Frequency vs Churn — Conclusion

- **Correlation**:
  The Spearman correlation is **-0.151**, showing a **negative relationship** between login frequency and churn risk.
  → Customers who log in more frequently are slightly less likely to churn.

- **Churn by Frequency Buckets**:

  - **Q1 (lowest frequency)** → Churn rate ~63.5% (highest risk).
  - **Q5 (highest frequency)** → Churn rate ~43.1% (lowest risk).
  - Churn rate steadily declines from **low-frequency to high-frequency users**.
- **Overall Insight**:
  Customers who log in more often tend to stay engaged and are **less likely to churn**, while infrequent logins are a strong indicator of churn risk.

Encouraging **more frequent customer logins** (e.g., engagement campaigns, gamification, or reminders) could help reduce churn.

Q7. Which preferred offer type is best for customer satisfaction?

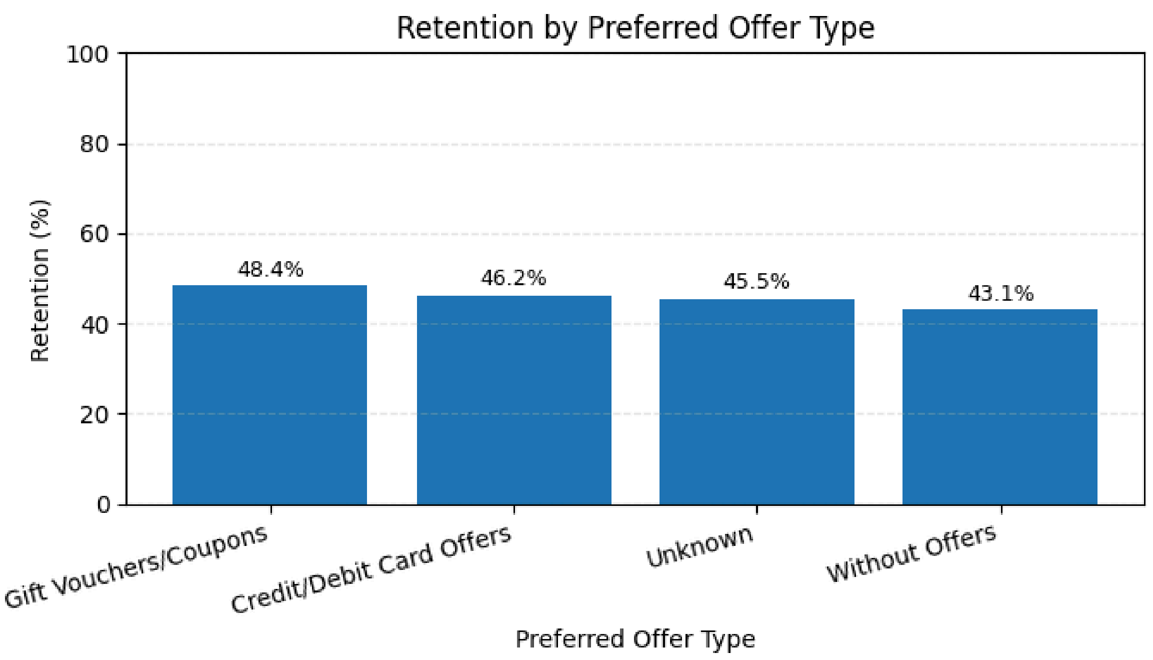
In [44]:
```
# Simple, cleaner chart with labels
ret = (1 - df.groupby("preferred_offer_types")["churn_risk_score"].mean()
ret = ret.round(2).sort_values(ascending=False)

plt.figure(figsize=(7,4))
```

```
bars = plt.bar(ret.index, ret.values)
plt.title("Retention by Preferred Offer Type")
plt.ylabel("Retention (%)");
plt.xlabel("Preferred Offer Type")
plt.ylim(0, 100);
plt.grid(axis="y", ls="--", alpha=0.3)

# add value labels
for b in bars:
    plt.text(b.get_x()+b.get_width()/2, b.get_height()+1, f"{b.get_height
             ha="center", va="bottom", fontsize=9)

plt.xticks(rotation=15, ha="right")
plt.tight_layout();
plt.show()
```



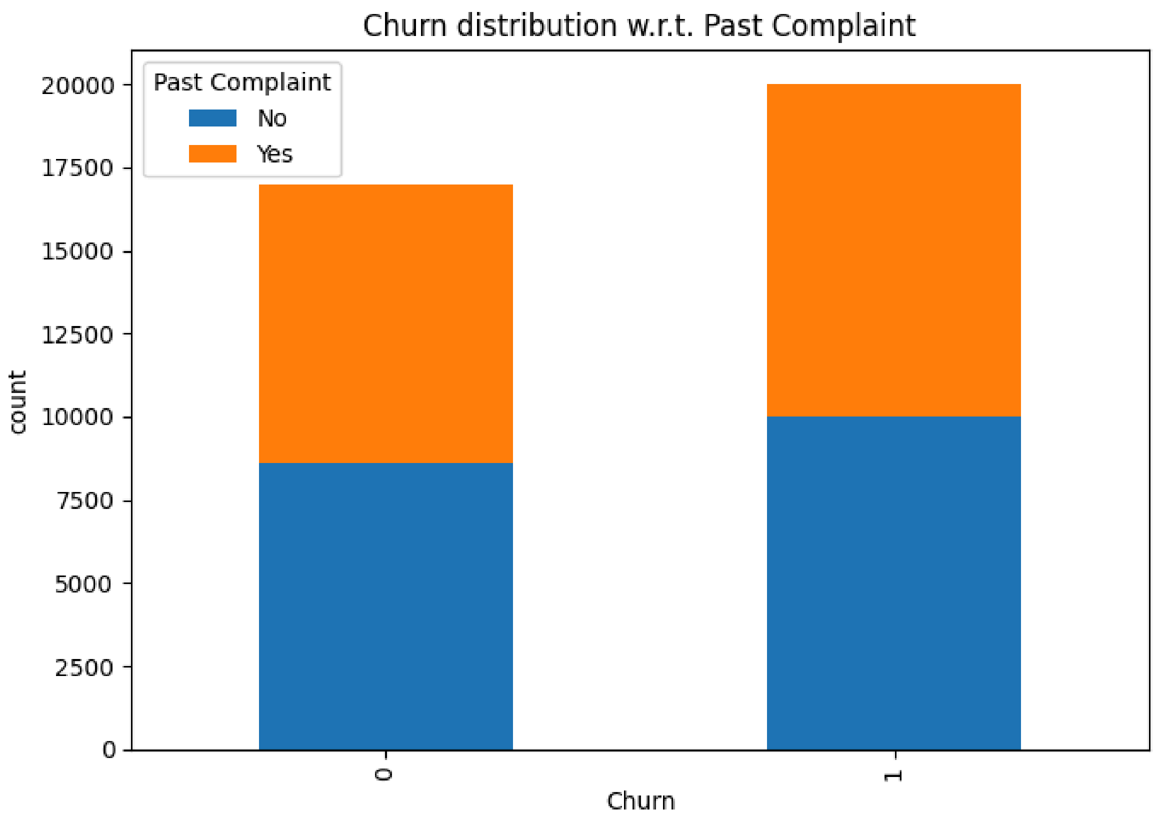
Preferred Offer Type vs Churn - Conclusion

- The gap is **modest (~5 pp)** from best (Vouchers) to lowest (No offers).
- This may be **statistically small** and **operationally marginal**.

Q8. Do customers with past complaints churn more?

In [45]:
```
ct_counts = pd.crosstab(df["churn_risk_score"], df["past_complaint"])  #
ax = ct_counts.plot(kind="bar", stacked=True, figsize=(7,5))

ax.set_title("Churn distribution w.r.t. Past Complaint")
ax.set_xlabel("Churn")
ax.set_ylabel("count")
ax.legend(title="Past Complaint")
plt.tight_layout()
plt.show()
```

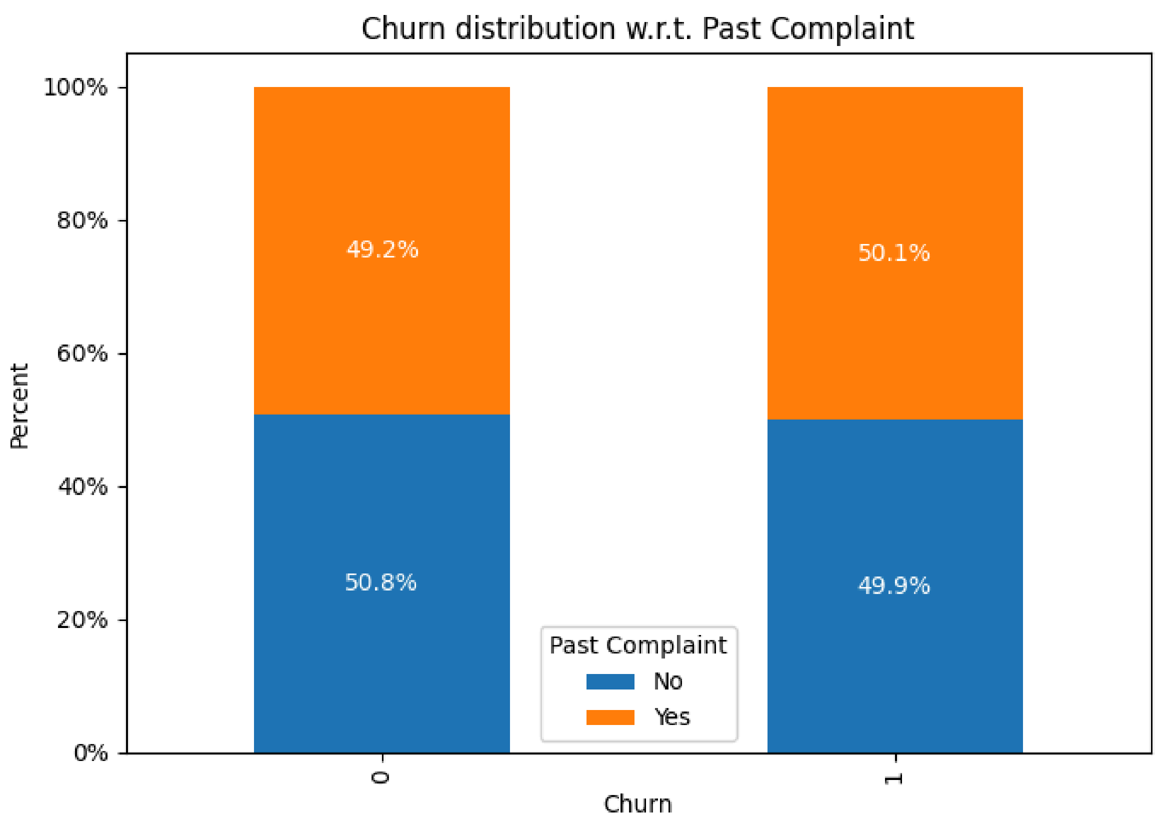


In [46]:
```
# proportions per churn group
ct = pd.crosstab(df["churn_risk_score"], df["past_complaint"], normalize=

ax = ct.plot(kind="bar", stacked=True, figsize=(7,5))
ax.set_title("Churn distribution w.r.t. Past Complaint")
ax.set_xlabel("Churn")
ax.set_ylabel("Percent")
ax.yaxis.set_major_formatter(PercentFormatter(1.0))  # 0–1 → %

# add % labels on each stacked segment
for p in ax.patches:
    width, height = p.get_width(), p.get_height()
    if height > 0:
        x = p.get_x() + width/2
        y = p.get_y() + height/2
        ax.text(x, y, f"{height*100:.1f}%", ha="center", va="center", fon

ax.legend(title="Past Complaint")
plt.tight_layout()
plt.show()
```



Past Complaints → Churn (Conclusion)

- Churn shares are ~50/50 for both groups → **past complaints alone don't drive churn**.
- Likely explanation: **post-complaint support is effective**, neutralizing risk.

Recommendation: Maintain/scale the current service-recovery playbook (fast SLA, follow-ups). Use complaints as a **trigger** to offer loyalty perks or onboarding tips, but prioritize other risk signals (low usage, entry tiers, non-referral) for retention targeting.

Q9. (Optional) Which features (customer behaviors) have most impacts churn vs not-churn? (RandomForest + SHAP value)

In [47]:
```
# y
y = df["churn_risk_score"].astype(int)
```

In [48]:
```
# Split features into numeric vs categorical
cat_cols = ['gender', 'region_category', 'membership_category', 'joined_t
             'medium_of_operation', 'internet_option', 'used_special_disco
             'past_complaint', 'complaint_status', 'feedback']
num_cols = ['age', 'avg_frequency_login_days', 'avg_time_spent', 'avg_tra
              'days_since_last_login', 'joining_date', 'points_in_wallet']
datetime_cols = ["joining_date"]
```

In [49]:
```
X_num = df[num_cols].apply(pd.to_numeric, errors="coerce")
```

```
In [50]: X_cat = pd.get_dummies(df[cat_cols], drop_first=True)
```

```
In [51]: # Set the time when I experience this code is now, you can consider it as
         now = pd.to_datetime("20-09-2025", format="%d-%m-%Y")
```

```
In [52]: X_dt = pd.DataFrame({
             f"{c}_age_days": (
                 (now - pd.to_datetime(df[c], errors="coerce", infer_datetime_form
                 .dt.total_seconds() / 86400
             ).clip(lower=0)
             for c in datetime_cols
         })
```

```
/var/folders/6x/h489kd6s4lxbdnxrhw8mvt140000gn/T/ipykernel_44140/281257301
9.py:3: UserWarning: The argument 'infer_datetime_format' is deprecated an
d will be removed in a future version. A strict version of it is now the d
efault, see https://pandas.pydata.org/pdeps/0004-consistent-to-datetime-pa
rsing.html. You can safely remove this argument.
  (now - pd.to_datetime(df[c], errors="coerce", infer_datetime_format=Tru
e))
```

```
In [53]: X = pd.concat([X_num, X_cat, X_dt], axis=1).fillna(0)
```

```
In [54]: Xtr, Xte, ytr, yte = train_test_split(X, y, test_size=0.2, stratify=y, ra
```

```
In [55]: # --- make sure Xtr, Xte are numeric ---
         Xtr_num = (Xtr.apply(pd.to_numeric, errors="coerce")
                       .fillna(0)
                       .astype("float64"))

         Xte_num = (Xte.reindex(columns=Xtr_num.columns)   # keep same column orde
                       .apply(pd.to_numeric, errors="coerce")
                       .fillna(0)
                       .astype("float64"))

         assert list(Xtr_num.columns) == list(Xte_num.columns)
```

```
In [56]: # Train RandomForest
         clf = RandomForestClassifier(n_estimators=200, random_state=42)
         clf.fit(Xtr_num, ytr)
```

```
Out[56]: ▾ RandomForestClassifier
         RandomForestClassifier(n_estimators=200, random_state=42)
```

```
In [ ]: explainer = shap.Explainer(
            clf, Xtr_num,
            model_output="probability",          # we're explaining proba
            feature_names=Xtr_num.columns
        )

        # returns an Explanation
        sh = explainer(Xte_num, check_additivity=False)
```

```
 91%|==================   | 13430/14798 [11:42<01:11]
```

In [ ]:
```
# slice class - 1 (churn)
sh_churn = sh[..., 1]

# bar plot of top 15 features
shap.plots.bar(sh_churn, max_display=15)
```

Key Features Driving Churn (SHAP)

- **Wallet balance** → stronge factor; higher balance = lower churn.
- **Membership tier** → Platinum/Premium/Gold reduce churn; No/Silver increase it.
- **Spending** (`avg_transaction_value`) → moderate impact; higher spenders churn less.
- **Engagement** (login frequency) → small but consistent; frequent users churn less.
- **Tenure & demographics** → minimal effect.

Recommendation: Focus retention on **wallet incentives** and **membership upgrades**, with re-engagement for low-activity users.

Q10. (Optional) How many separate customer groups based on their behaviors? (Kmean Clustering)

In [ ]:
```
# Scale (clustering works better in standardized space)
scaler = StandardScaler()
X_std = pd.DataFrame(scaler.fit_transform(X), columns=X.columns, index=X.
```

In [ ]:
```
K = range(2, 9)
inertias = []
models = []

for k in K:
    km = KMeans(n_clusters=k, random_state=42, n_init="auto")
    km.fit(X_std)
    inertias.append(km.inertia_)  # sum of squared distances (SSE)
    models.append(km)

plt.figure(figsize=(6,4))
plt.plot(list(K), inertias, "-o")
plt.xticks(list(K))
plt.xlabel("Number of clusters (k)")
plt.ylabel("Inertia / SSE")
plt.title("Elbow Method")
plt.grid(True, alpha=0.3)
plt.show()
```

Elbow Method Conclusion

The curve bends around **k=4–5**, meaning the optimal number of clusters is likely **4** (simple) or **5** (more detail).

In [ ]:
```
best_k = 4
cluster_labels = models[best_k-1].labels_            # integers 0..k-1
```

```
print(cluster_labels)
xy = PCA(n_components=2, random_state=42).fit_transform(X_std)

plt.figure(figsize=(6,5))
plt.scatter(xy[:,0], xy[:,1], c=cluster_labels, s=8, cmap="tab10")
plt.title("Customer Segments (PCA view)")
plt.xlabel("PC1"); plt.ylabel("PC2")
plt.show()
```

In [ ]:
```
best_k = 5
cluster_labels = models[best_k-1].labels_            # integers 0..k-1
print(cluster_labels)
xy = PCA(n_components=2, random_state=42).fit_transform(X_std)

plt.figure(figsize=(6,5))
plt.scatter(xy[:,0], xy[:,1], c=cluster_labels, s=8, cmap="tab10")
plt.title("Customer Segments (PCA view)")
plt.xlabel("PC1"); plt.ylabel("PC2")
plt.show()
```

Conclusion: Best K for Customer Segmentation

From the Elbow method, both **k=4** and **k=5** looked plausible.
However, the PCA scatter plots show that with **k=4**, clusters are overlapping and not well separated.
With **k=5**, the clusters are **clearer and more distinct**, suggesting that the optimal number of customer groups is **5**.

In [ ]:
```
best_k = 5
best_model = models[best_k-1]
```

In [ ]:
```
# === Most distinctive behaviors per cluster (top features) ===
C = best_model.cluster_centers_

# z-scores across clusters (per feature/column)
Z = (C - C.mean(axis=0)) / (C.std(axis=0) + 1e-9)

# feature names used to fit KMeans
feat_names = X.columns.to_list()

top_n  = 5
for c in range(best_k):
    # indices of the largest absolute z-scores for cluster c
    idx = np.argsort(-np.abs(Z[c]))[:top_n]
    print(f"\nCluster {c}: top behaviors (z-score sign shows direction)")
    for i in idx:
        direction = "higher" if Z[c, i] > 0 else "lower"
        print(f"  - {feat_names[i]}: much {direction} than the average of
```

K-Means Segmentation — Conclusion

How many groups?
Using the elbow method and PCA separation, the data are best segmented into **5 distinct customer groups**.

High-level profiles (top behaviors per cluster)

- **Cluster 0 – Mobile & frequent users (Village-leaning)**
    - Much **higher** mobile-data internet usage and **higher login frequency**.
    - **Lower time spent** per session; **more Village** region.
    - Dislike "**Without Offers**" (prefer to have some kind of offer available).
- **Cluster 1 – Issues solved via follow-up (female-leaning)**
    - Complaints and feedback **solved in follow-up** far **higher** than average.
    - **Referral source unknown** is higher; **male share lower**.
    - **Lower** interest in **Gift Vouchers/Coupons**.
- **Cluster 2 – Older, established members with few complaints**
    - **Older age**; **fewer "No Membership"** (i.e., more are members).
    - Complaint/feedback often **"Not Applicable"**; **past complaints lower**.
- **Cluster 3 – Quick resolutions & offer-active**
    - **Solved** complaints/feedback **much higher**.
    - **Less** "joined by referral"; **fewer Platinum** members.
    - **Higher** willingness to **apply for offers**.
- **Cluster 4 – Younger, offer-applicants but discount-light**
    - **Younger age**.
    - **Less** use of **special discounts**, yet **more** likely to **apply for offers**.
    - Complaint/feedback often **"Not Applicable"**.

Takeaway:
Five segments provide clear, behaviorally distinct groups useful for targeted offers, complaint-handling strategies, and channel personalization.

8) Conclusion

Final Conclusion & Business Recommendations

Summary of Findings

- **Membership tier is the strongest churn driver**: Low tiers (No/Silver/Basic) have high churn; Premium/Platinum show strong loyalty.
- **Financial behavior matters**: Higher wallet balances reduce churn; high-spending customers are more churn-sensitive.
- **Engagement reduces churn**: Frequent logins lower churn risk; infrequent logins are a warning signal.
- **Complaints are not churn drivers** if resolved quickly—service recovery is effective.

- **Demographics (gender, offers)** play a minor role, but offer type still influences behavior.
- **Best segmentation = 5 clusters** with distinct profiles (mobile-heavy users, follow-up resolution seekers, older loyal members, quick-resolution/offer users, younger discount-light users).

Business Decision-Making

1. **Retention & Loyalty Programs**
    - Push **wallet incentives and tier upgrades** for No/Silver/Basic members.
    - Protect high-value Premium/Platinum members with **exclusive perks**.
2. **Customer Engagement**
    - Increase logins via **gamification, reminders, app campaigns**.
    - For low-frequency users, design **reactivation offers**.
3. **Complaint Management**
    - Keep **fast SLA and follow-up processes**, as they neutralize churn risk.
    - Use complaints as a **trigger for loyalty outreach**.
4. **Pricing & Offers**
    - Manage high-spender churn with **targeted discounts, premium services, or loyalty rewards**.
    - Segment offers: vouchers for price-sensitive clusters, personalized deals for offer-active clusters.
5. **Segment-Specific Strategies**
    - **Cluster 0 (Mobile & frequent users)** → Push app-based offers and regional campaigns.
    - **Cluster 1 (Follow-up resolution seekers)** → Emphasize **customer care quality**.
    - **Cluster 2 (Older loyal members)** → Build **long-term value programs**.
    - **Cluster 3 (Quick-resolution, offer-active)** → Upsell through **time-limited offers**.
    - **Cluster 4 (Young, discount-light)** → Promote **innovative bundles instead of heavy discounts**.

Takeaway

Churn is best reduced by **upgrading entry-tier members, incentivizing wallet balance, and boosting engagement**.
The **5 behavioral clusters** enable **personalized marketing and service strategies**, turning insights into **actionable business decisions**.

```
In [ ]:
```