Session 4.2: AI Applications in Business Decision-Making Part 1: Predictive Customer Analytics and Churn

THE BUSINESS SITUATION

- A subscription telecom company loses customers every month to churn (customers cancelling their contract)
- Its retention team can call customers or assign them an account manager, but every contact costs money, and calling customers who would stay anyway wastes it
- The head of customer retention asks: which of our current customers should we contact first, and will the campaign pay for itself? Today, you turn churn predictions into revenue at risk and a campaign plan with a measurable return

Today's flow: 60 min lecture, then 120 min hands-on lab

EXPECTED OUTCOMES

By the end of today, you will be able to:

1. ***"Who is likely to leave?"***
   Train a churn model on past customers and score current customers with a churn probability
2. ***"How much money is at stake?"***
   Calculate revenue at risk (churn probability x yearly customer value) and rank customers by it
3. ***"Will the campaign pay off?"***
   Calculate a campaign's expected net benefit and ROI (Return on Investment), choose who to target, and fix a `ValueError` caused by missing feature columns

Session Map: Follow the Numbers

Today's question: *Which of our current customers should we contact first, and will the campaign pay for itself?*

1 Why it matters
Lecture, part 1

1. **Session Purpose**
   answer today's question
2. **Learning Outcomes**
   3 skills you will show today
3. **Business Context**
   Pricing a Retention Campaign
4. **Data Basics**
   Learn from the Past, Act on the Present

→

2 Learn the tools
Lecture, part 2

5. **Lab Dataset**
   2 CSVs
6. **Keywords**
   6 terms
7. **Core Foundations**
   4 ideas, one at a time
8. **IDE Setup**
   name files, run in the Terminal
9. **Code Examples**
   3 demos + Quick Checks
10. **References**
    3 official docs

→

3 Practice
Lab

11. **Exercise 1: Guided**
    Campaign Business Case
12. **Exercise 2: Applied**
    Highest ROI or Most Money?
13. **Exercise 3: Advanced**
    The Best Offer for Each Customer

Pricing a Retention Campaign

KEY POINTS

- Four ideas, one example (current customer `ACT-101`):
- CLV (Customer Lifetime Value) is the total value of a customer over the whole relationship, often several years.
- A campaign is worth it for a customer when expected saved revenue is larger than the contact cost.
- Net benefit vs ROI: net benefit = saved revenue - cost, in dollars; ROI = net benefit / cost x 100, in percent.

SEE IT ON REAL DATA

Term	Meaning	Example
Churn probability	the model's estimate that this customer leaves within 60 days	`0.876`
Annual value	monthly fee x 12: one year of revenue from this customer	`$2,399.88`
Revenue at risk	churn probability x annual value: the revenue we expect to lose	`$2,102.29`
Expected saved revenue	revenue at risk x success rate of the campaign	`$2,102.29 x 0.40 = $840.92`

Learn from the Past, Act on the Present

KEY POINTS

- Two groups of customers, two files:
- A retention campaign only makes sense for customers who have not left yet.
- What drives churn in our history: after scaling, the model's coefficients are `-0.718` for tenure (longer customers stay), `+0.897` for support calls, and `+0.878` for usage drop (more calls and falling usage signal leaving).

AT A GLANCE

File	Customers	Has the answer (`Churned`)?	Used to
History	14 past customers	yes	train the model
Active	10 current customers	no, not yet known	score and target

The Data for Today

FILES

- `data/session_04_02_dataset.csv`
- `data/session_04_02_active.csv`

Columns:

COLUMNS

`Subscriber_ID`	Customer code (`SUB-...` in the history, `ACT-...` for active customers).
`Tenure_Months_Count`	Months as a customer.
`Monthly_Subscription_Fee_USD`	Monthly fee in U.S. Dollars.
`Customer_Support_Calls_Count`	Calls to customer support in the last 90 days.
`Usage_Drop_Last_30_Days_Pct`	How much the customer's use of the service fell in the last 30 days, in percent.

6 Terms You Need Today

TERM 1

`Churn`

What: customers ending their relationship with the company

Why: in subscription businesses, keeping a customer is usually cheaper than winning a new one

TERM 2

`Churn Probability`

What: the model's estimate, from 0 to 1, that a customer will leave

Why: ranks customers by risk instead of treating everyone the same

TERM 3

`Revenue at Risk`

What: churn probability x annual value

Why: turns a probability into dollars a manager can budget against

TERM 4

`Customer Lifetime Value (CLV)`

What: the total value of a customer over the whole relationship

Why: decides how much a customer is worth saving

TERM 5

`Net Benefit` and `ROI`

What: saved revenue minus cost (dollars), and net benefit divided by cost (percent)

Why: the two numbers a finance team asks for before approving a campaign

TERM 6

`Targeting Threshold`

What: the minimum churn probability for a customer to be contacted

Why: sets the size and cost of the campaign

How It Works

7.1

Learn from the past, predict the present

Remember: fit on the past, score the present.

7.2

Expected value

Remember: revenue at risk = probability x value; rank by dollars.

7.3

The economics of a campaign

Remember: contact a customer only when expected savings exceed the contact cost.

7.4

Matching the offer to the customer

Remember: set a default, then overwrite from the cheapest offer to the most valuable.

7.1 Learn from the past, predict the present: On Real Data

SEE IT ON REAL DATA

Average of past customers	Stayed (9)	Left (5)
Months as a customer	36.9	6.8
Support calls (last 90 days)	1.1	4.6
Usage drop (last 30 days)	9.4%	46.0%

Remember: fit on the past, score the present.

Learn from the Past, Score the Present

YourName_S42_demo_1.py — Input → Process → Output

```
# 1. Input: past customers (with outcomes) and current customers
import pandas as pd
from sklearn.pipeline import Pipeline
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LogisticRegression
history = pd.read_csv("data/session_04_02_dataset.csv")
active = pd.read_csv("data/session_04_02_active.csv")
features = ["Tenure_Months_Count", "Customer_Support_Calls_Count",
            "Usage_Drop_Last_30_Days_Pct"]

# 2. Process: train on history, then score the active customers
model = Pipeline([
    ("scale", StandardScaler()),
    ("classify", LogisticRegression()),
])
model.fit(history[features], history["Churned_Within_60_Days_Binary"])
active["Churn_Probability"] = model.predict_proba(active[features])[:, 1].round(3)

# 3. Output: the active customers, most at risk first
print(active.sort_values("Churn_Probability", ascending=False)[
    ["Subscriber_ID", "Tenure_Months_Count", "Churn_Probability"]
])
```

HOW IT WORKS

- Two files are loaded: `history` (past customers, with the answer) and `active` (current customers, without it).
- The `Pipeline` scales the 3 features and fits a logistic regression on the history (section 7.1).
- `model.predict_proba(active[features])[:, 1]` scores the current customers: the second column is the probability of leaving.

READ THE OUTPUT

- The top 3 have all been customers for less than 8 months, like the past customers who left.

Quick Check 1: `named_steps` and `.coef_`

NEW TOOL

`named_steps` and `.coef_`

`model.named_steps["classify"].coef_` returns the learned coefficients of the logistic regression inside the pipeline. Why: shows which signals push a customer toward leaving.

YOUR TASK

Fill in the Blank

Add one line to part **3. Output**: `print(dict(zip(features, model.named_steps["classify"].____[0].round(3).tolist( ))))`

Check: tenure `-0.718`, support calls `0.897`, usage drop `0.878`. A negative coefficient lowers the churn probability; a positive one raises it.

Revenue at Risk

YourName_S42_demo_2.py — Input → Process → Output

```
# 1. Input: train and score, as in Micro-Example 1
import pandas as pd
from sklearn.pipeline import Pipeline
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LogisticRegression
history = pd.read_csv("data/session_04_02_dataset.csv")
active = pd.read_csv("data/session_04_02_active.csv")
features = ["Tenure_Months_Count", "Customer_Support_Calls_Count",
            "Usage_Drop_Last_30_Days_Pct"]
model = Pipeline([("scale", StandardScaler()),
                  ("classify", LogisticRegression())])
model.fit(history[features], history["Churned_Within_60_Days_Binary"])
active["Churn_Probability"] = model.predict_proba(active[features])[:, 1].round(3)

# 2. Process: one year of value, and the part of it at risk
active["Annual_Value_USD"] = (active["Monthly_Subscription_Fee_USD"] * 12).round(2)
active["Revenue_At_Risk_USD"] = (
    active["Churn_Probability"] * active["Annual_Value_USD"]
).round(2)
ranked = active.sort_values("Revenue_At_Risk_USD", ascending=False)

# 3. Output: the ranking and the total
print(ranked[["Subscriber_ID", "Churn_Probability", "Annual_Value_USD",
              "Revenue_At_Risk_USD"]])
print(f"Total revenue at risk: ${active['Revenue_At_Risk_USD'].sum():,.2f}")
```

HOW IT WORKS

- Part 1 repeats the training and scoring of Micro-Example 1.
- `Monthly_Subscription_Fee_USD * 12` is one year of revenue from each customer.
- Multiplying by the churn probability gives the revenue at risk (section 7.2).

READ THE OUTPUT

- The order differs from Micro-Example 1: `ACT-101` moves to the top because it pays the most, and `ACT-110` jumps to fourth despite its lower probability.

Quick Check 2: `.cumsum()`

NEW TOOL

`.cumsum()`

`column.cumsum()` gives a running total down the rows. Why: shows how much of the total risk the first few customers carry.

YOUR TASK

Fill in the Blank

Add one line at the end of part **2. Process**: `ranked["Running_Total_USD"] = ranked["Revenue_At_Risk_USD"].______().round(2)`, and add `"Running_Total_USD"` to the printed columns.

Check: the third row shows `4851.27`: the top 3 customers carry 63% of the total risk.

Assign a Campaign to Each Customer

YourName_S42_demo_3.py — Input → Process → Output

```
# 1. Input: train and score, as in Micro-Example 2
import pandas as pd
from sklearn.pipeline import Pipeline
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LogisticRegression
history = pd.read_csv("data/session_04_02_dataset.csv")
active = pd.read_csv("data/session_04_02_active.csv")
features = ["Tenure_Months_Count", "Customer_Support_Calls_Count",
            "Usage_Drop_Last_30_Days_Pct"]
model = Pipeline([("scale", StandardScaler()),
                  ("classify", LogisticRegression())])
model.fit(history[features], history["Churned_Within_60_Days_Binary"])
active["Churn_Probability"] = model.predict_proba(active[features])[:, 1].round(3)
active["Annual_Value_USD"] = (active["Monthly_Subscription_Fee_USD"] * 12).round(2)

# 2. Process: rules from the cheapest to the most valuable campaign
active["Campaign"] = "Newsletter"
active.loc[active["Churn_Probability"] >= 0.4, "Campaign"] = "Retention_Call"
active.loc[
    (active["Churn_Probability"] >= 0.7) & (active["Annual_Value_USD"] >= 1500),
    "Campaign",
] = "VIP_Account_Manager"

# 3. Output: how many customers get each campaign
print(active["Campaign"].value_counts())
print(active[["Subscriber_ID", "Churn_Probability", "Campaign"]])
```

HOW IT WORKS

- Part 1 scores the customers and computes their annual value, as before.
- `active["Campaign"] = "Newsletter"` gives everyone the default.
- The first `.loc[]` line overwrites the default with "Retention_Call" for customers at `0.4` or above.

READ THE OUTPUT

```
Campaign
Newsletter              4
Retention_Call          4
...
```

- 2 customers get a personal account manager, 4 get a call, and 4 only get the newsletter.

Quick Check 3: `pd.cut()`

NEW TOOL

`pd.cut()`

`pd.cut(column, bins=[0, 0.4, 0.7, 1.0], labels=["Low", "Medium", "High"])` puts each value into a band. Why: managers think in risk bands, not in decimals.

YOUR TASK

Fill in the Blank

Add two lines to part **3. Output**: `active["Risk_Band"] = pd.___(active["Churn_Probability"], bins=[0, 0.4, 0.7, 1.0], labels=["Low", "Medium", "High"])` and `print(active["Risk_Band"].value_counts())`

Check: `Low` `4`, `Medium` `3`, `High` `3`.

Where to Look Things Up

REFERENCE

Scikit-Learn Logistic Regression: predict_proba Documentation

https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html#sklearn.linear_model.LogisticRegression.predict_proba

REFERENCE

Pandas API Reference: pandas.cut

https://pandas.pydata.org/docs/reference/api/pandas.cut.html

REFERENCE

Fader & Hardie: Probability Models for Customer-Base Analysis (Wharton Marketing Standards)

https://www.brucehardie.com/papers/

Three Exercises

EXERCISE 1: GUIDED

Campaign Business Case

Goal: Price a campaign for customers at `0.5` or above

File: `YourName_S42_churn.py`

Check: net benefit `$1,899.68`; ROI `633.2`%

EXERCISE 2: APPLIED

Highest ROI or Most Money?

Goal: Highest ROI or most money? Test thresholds 0.3 to 0.8

File: `YourName_S42_practice.py`

Check: best ROI at `0.8`; most money at `0.4`

EXERCISE 3: ADVANCED

The Best Offer for Each Customer

Goal: Choose the best offer for each customer

File: `YourName_S42_challenge.py`

Check: VIP for 4, Call for 3, None for 3

Full steps for all three exercises: sections 11-13 of the course page.