Session 3.1: Foundations of Artificial Intelligence: Enterprise Value Creation and Decision Automation

THE BUSINESS SITUATION

- Five departments have proposed 10 AI projects, costing between $80,000 and $220,000 each
- The total request is $1.5 million, but this year's AI budget is only $500,000
- The COO (Chief Operating Officer) asks: which AI projects should we fund with $500,000, and how fast will they pay back? Today, you learn what AI can and cannot do, and use Python to compare the proposals on the same financial terms

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. ***"Is this project really AI?"***
   Tell apart AI (Artificial Intelligence), ML (Machine Learning), and DL (Deep Learning), and decide when a simple rule is enough
2. ***"Which projects pay back fastest?"***
   Calculate the payback period (months until the savings repay the cost) and the 3-year ROI (Return on Investment)
3. ***"Which projects fit in $500,000?"***
   Allocate a fixed budget with a loop, and fix a `ZeroDivisionError` (dividing by zero)

Session Map: Follow the Numbers

Today's question: *Which AI projects should we fund with $500,000, and how fast will they pay back?*

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**
   How to Judge an AI Project
4. **Data Basics**
   AI, ML, DL, and When a Rule Is Enough

→

2 Learn the tools
Lecture, part 2

5. **Lab Dataset**
   1 CSV
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**
    4 official docs

→

3 Practice
Lab

11. **Exercise 1: Guided**
    AI Portfolio Ranking
12. **Exercise 2: Applied**
    Fund Projects Within $500,000
13. **Exercise 3: Advanced**
    Is the Greedy Choice the Best Choice?

How to Judge an AI Project

KEY POINTS

- Every AI proposal is an investment: money now, savings later.
- Savings are forecasts, not facts.
- Labor hours saved are not the same as money saved: hours only become savings if the freed time is used for other valuable work.

SEE IT ON REAL DATA

Term	Meaning	Example
Implementation cost	one-time cost to build the system	`150,000`
Running cost	yearly cost to keep it working (cloud computing, updates); here 15% of the build cost	`22,500`
Net annual benefit	yearly savings minus the running cost	`294,000 - 22,500 = 271,500`
Payback period	months until the benefit repays the build cost: cost / net benefit x 12	`6.6` months
3-year ROI	(net benefit x 3 - cost) / cost x 100	`443.0`%

AI, ML, DL, and When a Rule Is Enough

KEY POINTS

- Three nested ideas:
- Rule or ML?
  - **Use a rule** when the policy is fixed and written down: "sales tax is 5% of the price". A rule is cheaper, exact, and easy to explain.
  - **Use ML** when the pattern is complex or changes over time, and you have many past examples with known outcomes: "which invoices look like fraud?"
- ML makes a prediction; a person or a business rule still decides what to do with it.

AT A GLANCE

Term	What it is	Business example
AI (Artificial Intelligence)	any computer system that does a task we link to human thinking	a chatbot answering customers
ML (Machine Learning)	AI that **learns a rule from past examples** instead of being told the rule	flagging likely fraud from past fraud cases
DL (Deep Learning)	ML with many-layered neural networks, for images, sound, and text	reading scanned invoices

The Data for Today

FILES

- `data/session_03_01_dataset.csv`

Audited 10-row AI project proposal dataset (fictional teaching data), one row per proposal:

COLUMNS

`Project_ID`	Proposal code (e.g., `AI-PRJ-101`).
`Department_Name`	Sponsoring department (Supply Chain, Customer Service, Marketing, Finance, Fraud Operations).
`Implementation_Cost_USD`	One-time build cost in U.S. Dollars.
`Annual_Labor_Hours_Saved_Count`	Staff hours the system is expected to save per year.
`Projected_Cost_Savings_USD`	Expected yearly savings in U.S. Dollars.
`Payback_Period_Months_Count`	The department's own payback estimate. It ignores running costs, so it is shorter than the payback you calculate today (for example, `6.1` against `6.6` months for `AI-PRJ-101`).

6 Terms You Need Today

TERM 1

`Artificial Intelligence (AI)`

What: computer systems that do tasks we link to human thinking

Why: the umbrella term in every vendor pitch; ask which kind of AI is inside

TERM 2

`Machine Learning (ML)`

What: a system that learns a decision rule from past examples

Why: handles patterns too complex to write down by hand

TERM 3

`Deep Learning (DL)`

What: ML with many-layered neural networks

Why: needed for images, sound, and free text; costly to build and run

TERM 4

`Rule-Based System`

What: code that follows rules a person wrote, such as `if amount > 250000:`

Why: the right choice when the policy is clear and fixed

TERM 5

`Payback Period`

What: months until the net benefit repays the build cost

Why: shows how fast the money comes back

TERM 6

`ROI (Return on Investment)`

What: total gain compared with the cost, in percent

Why: compares projects of different sizes on one scale

How It Works

7.1

Rule or learning

Remember: clear, fixed policy: write a rule; complex pattern with many past answers: learn it.

7.2

Payback and ROI

Remember: same formula for every project, running costs included.

7.3

Division by zero

Remember: a zero in the bottom of a fraction must be caught, not ignored.

7.4

Choosing under a budget

Remember: greedy is fast and simple; with few options, check the combinations too.

7.1 Rule or learning: On Real Data

SEE IT ON REAL DATA

Approach	The rule	Mistakes on the 10 invoices
Rule-based	a person chose "above 250,000"	`1` (the honest `260000` invoice is flagged)
Learned	the search tried every amount as a threshold and found "300,000 or more"	`0`

Remember: clear, fixed policy: write a rule; complex pattern with many past answers: learn it.

Payback and ROI for Every Proposal

YourName_S31_demo_1.py — Input → Process → Output

```
# 1. Input: load the 10 proposals
import pandas as pd
df = pd.read_csv("data/session_03_01_dataset.csv")

# 2. Process: net benefit, payback, and 3-year ROI
df["Net_Annual_Benefit_USD"] = (
    df["Projected_Cost_Savings_USD"] - df["Implementation_Cost_USD"] * 0.15
)
df["Payback_Months"] = (
    df["Implementation_Cost_USD"] / df["Net_Annual_Benefit_USD"] * 12
).round(1)
df["ROI_3yr_Pct"] = (
    (df["Net_Annual_Benefit_USD"] * 3 - df["Implementation_Cost_USD"])
    / df["Implementation_Cost_USD"] * 100
).round(1)

# 3. Output: the 3 proposals with the highest ROI
top3 = df.sort_values("ROI_3yr_Pct", ascending=False).head(3)
print(top3[["Project_ID", "Department_Name", "Payback_Months", "ROI_3yr_Pct"]])
```

HOW IT WORKS

- `df["Implementation_Cost_USD"] * 0.15` is the yearly running cost; subtracting it from the projected savings gives the net annual benefit, for all 10 proposals at once.
- The payback divides the build cost by the net benefit and multiplies by 12 to turn years into months.
- The 3-year ROI compares 3 years of net benefit with the build cost, in percent (section 7.2).

READ THE OUTPUT

- Both Fraud Operations projects pay back in under 5 months and return about 7 times their cost over 3 years.

Quick Check 1: `.rank()`

NEW TOOL

`.rank()`

`column.rank(ascending=False)` gives each row its place, 1 for the largest. Equal values share the average place, such as `4.5`. Why: a ranking column is easy to show in a decision memo.

YOUR TASK

Fill in the Blank

Add one line at the end of part **2. Process**: `df["ROI_Rank"] = df["ROI_3yr_Pct"].____(ascending=False)`, then add `"ROI_Rank"` to the printed columns.

Check: `AI-PRJ-110` has rank `1.0` and `AI-PRJ-105` has rank `2.0`.

A Fixed Rule vs a Rule Learned from Examples

YourName_S31_demo_2.py — Input → Process → Output

```
# 1. Input: 10 past invoices, checked by auditors (fictional)
import pandas as pd
past = pd.DataFrame({
    "Amount_USD": [120000, 180000, 210000, 240000, 260000,
                   300000, 330000, 410000, 450000, 520000],
    "Was_Fraud": [False] * 5 + [True] * 5,
})

# 2. Process: a rule a person chose vs a threshold learned from the data
rule_flags = past["Amount_USD"] > 250000
rule_mistakes = (rule_flags != past["Was_Fraud"]).sum()

best_threshold, best_mistakes = None, len(past)
for threshold in past["Amount_USD"]:
    flags = past["Amount_USD"] >= threshold
    mistakes = (flags != past["Was_Fraud"]).sum()
    if mistakes < best_mistakes:
        best_threshold, best_mistakes = threshold, mistakes

# 3. Output: compare the mistakes
print("Fixed rule (above 250,000) mistakes:", rule_mistakes)
print("Learned threshold:", best_threshold, "mistakes:", best_mistakes)
```

HOW IT WORKS

- `pd.DataFrame({...})` builds 10 fictional past invoices with known answers.
- `past["Amount_USD"] > 250000` applies the person's rule and gives one `True` / `False` per invoice.
- `(rule_flags != past["Was_Fraud"]).sum()` counts the rows where the rule and the auditors disagree: the mistakes.

READ THE OUTPUT

```
Fixed rule (above 250,000) mistakes: 1
Learned threshold: 300000 mistakes: 0
```

- The person's rule flags the honest `260000` invoice: 1 mistake.

Quick Check 2: `.mean()` of True/False

NEW TOOL

`.mean()` of True/False

`.mean()` on a `True` / `False` column gives the share of `True` values, because `True` counts as 1. Why: tells a manager what share of invoices a rule would send to review.

YOUR TASK

Fill in the Blank

Add one line to part **3. Output**: `print("Share flagged by the fixed rule:", rule_flags.____())`

Check: the Terminal shows `Share flagged by the fixed rule: 0.6`, so 6 of 10 invoices would go to review.

A Shortlist That Does Not Fit the Budget

YourName_S31_demo_3.py — Input → Process → Output

```
# 1. Input: load the proposals and add the payback period
import pandas as pd
df = pd.read_csv("data/session_03_01_dataset.csv")
df["Net_Annual_Benefit_USD"] = (
    df["Projected_Cost_Savings_USD"] - df["Implementation_Cost_USD"] * 0.15
)
df["Payback_Months"] = (
    df["Implementation_Cost_USD"] / df["Net_Annual_Benefit_USD"] * 12
).round(1)

# 2. Process: fast payback AND many hours saved
shortlist = df[
    (df["Payback_Months"] <= 7.0)
    & (df["Annual_Labor_Hours_Saved_Count"] >= 4000)
]
shortlist_cost = shortlist["Implementation_Cost_USD"].sum()

# 3. Output: compare the shortlist with the budget
print(shortlist[["Project_ID", "Department_Name", "Payback_Months"]])
print(f"Shortlist cost: ${shortlist_cost:,.0f} against a budget of $500,000")
```

HOW IT WORKS

- Part 1 recomputes the net benefit and payback, as in Micro-Example 1.
- The two conditions keep projects that pay back within 7 months **and** save at least 4,000 hours a year; each condition is in its own brackets, joined with `&` (Session 1.4).
- `shortlist["Implementation_Cost_USD"].sum()` adds up what the shortlist would cost.

READ THE OUTPUT

```
   Project_ID   Department_Name  Payback_Months
0  AI-PRJ-101      Supply Chain             6.6
3  AI-PRJ-104           Finance             6.6
...
```

- 6 projects pass both rules, but together they cost more than twice the budget.

Quick Check 3: `.isin()`

NEW TOOL

`.isin()`

`column.isin([...])` is `True` when the value is in the list. Why: answers questions about a few named groups, such as "show only Finance and Fraud Operations".

YOUR TASK

Fill in the Blank

Add one line to part **3. Output**:

```
print(len(df[df["Department_Name"].____(["Finance", "Fraud Operations"])]))
```

Check: the Terminal shows `4`.

Where to Look Things Up

REFERENCE

Scikit-Learn: Machine Learning in Python Official Overview

https://scikit-learn.org/stable/

REFERENCE

Russell & Norvig: Artificial Intelligence: A Modern Approach (Reference Standards)

https://aima.cs.berkeley.edu/

REFERENCE

Agrawal, Gans, Goldfarb: Prediction Machines: The Simple Economics of AI (Harvard Business Review Press)

https://www.predictionmachines.ai/

REFERENCE

Python Library: itertools.combinations

https://docs.python.org/3/library/itertools.html#itertools.combinations

Three Exercises

EXERCISE 1: GUIDED

AI Portfolio Ranking

Goal: Rank the AI portfolio and compare departments

File:
`YourName_S31_portfolio.py`

Check: Fraud Operations leads (`709.6`)

EXERCISE 2: APPLIED

Fund Projects Within $500,000

Goal: Fund projects by hours per dollar until $500,000 runs out

File:
`YourName_S31_practice.py`

Check: 4 projects, `$480,000`, `16,100` hours a year

EXERCISE 3: ADVANCED

Is the Greedy Choice the Best Choice?

Goal: Test the simple method against every combination

File:
`YourName_S31_challenge.py`

Check: `1023` groups checked, `163` fit the budget

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