Session 4.4: Ethical Considerations in Artificial Intelligence: Algorithmic Bias, Regulatory Compliance, and AI Governance

PEDAGOGICAL AGENDA

1:2 Delivery Rhythm

Session Learning Architecture

Structured 3-hour applied computing session designed for business students with zero prior coding experience.

- **Phase 1 (60 Min Lecture):** Computational foundations, business vocabulary, and syntactic mechanics.
- **Phase 2 (120 Min Lab):** Guided hands-on implementation, error diagnosis, and AI co-pilot resolution.
- **Zero Fluff Mandate:** Focus directly on executable Python code, tabular data wrangling, and commercial intelligence.

STRATEGIC OUTCOME

Operational Autonomy

Business Analytical Competency

Transforming passive spreadsheet users into autonomous programmatic decision makers.

- **Speed & Scale:** Processing datasets exceeding spreadsheet limits with instantaneous vectorized execution.
- **Repeatable Pipelines:** Replacing manual copy-paste workflows with fully automated Python scripts.
- **AI Debugging Literacy:** Leveraging modern IDEs to diagnose and resolve errors independently.

Core Computational Keywords & Business Meaning

CONCEPT 01 — Core Keyword

Algorithmic Disparate Impact

A legal and quantitative metric occurring when an automated decision system yields substantially different adverse outcomes across protected demographic groups.

CONCEPT 02 — Core Keyword

Demographic Parity Metric

The fairness standard requiring that the likelihood of receiving a positive algorithmic outcome (e.g., credit approval) is equal across all demographic groups.

CONCEPT 03 — Core Keyword

Model Explainability (SHAP / Feature Attribution)

Algorithmic frameworks that decompose complex black-box model predictions into transparent, auditable feature contributions for adverse action notices.

CONCEPT 04 — Core Keyword

Regulatory Compliance (EU AI Act Framework)

Auditing machine learning systems against emerging international governance standards (such as the European Union Artificial Intelligence Act for high-risk credit and hiring systems).

Computational Foundations & Business Operations

PILLAR 01 Architecture

**Core Programming & Computational Foundations

**

PILLAR 02 Architecture

Sources of Algorithmic Bias in Historical Training Data

Machine learning models do not learn ethics; they learn historical correlation patterns. If historical lending or hiring practices contained human bias, models trained on that data will codify and scale that discrimination with algorithmic efficiency.

PILLAR 03 Architecture

Quantitative Fairness Metrics (The Impossibility Theorem)

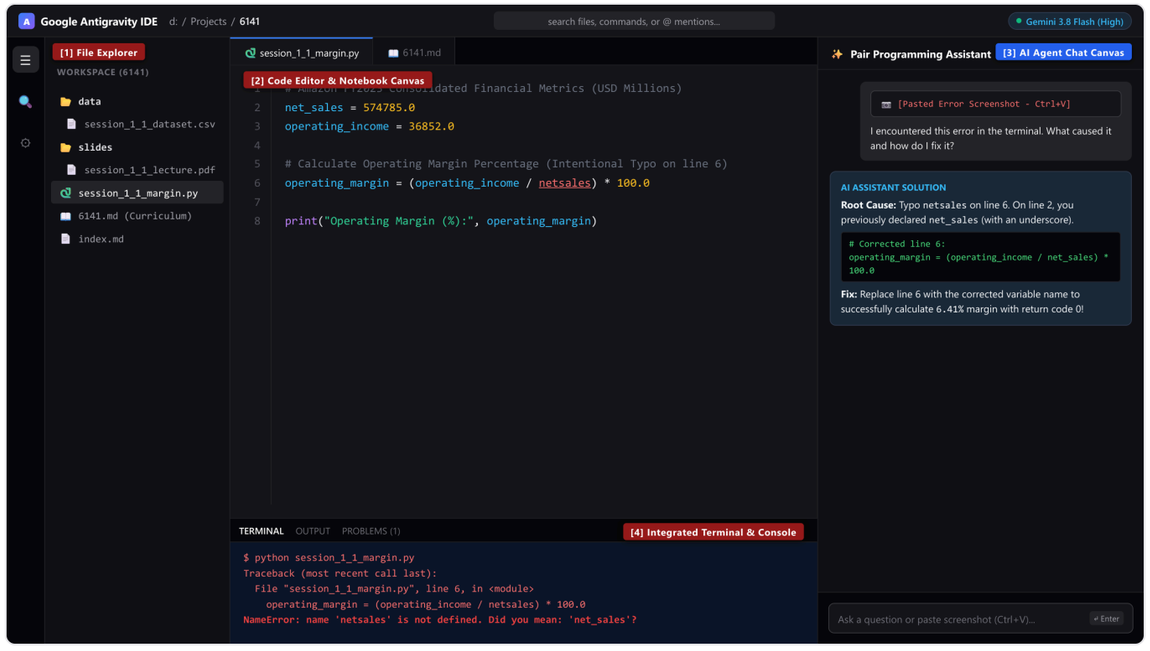
Understanding mathematical fairness trade-offs. It is mathematically impossible to satisfy Demographic Parity (equal approval rates), Equalized Odds (equal false positive rates), and Predictive Parity (equal accuracy) simultaneously when base demographic rates differ.

PILLAR 04 Architecture

The European Union Artificial Intelligence Act (EU AI Act)

High-risk automated systems (credit underwriting, hiring, insurance pricing, critical infrastructure) face strict legal requirements: mandatory technical documentation, human-in-the-loop oversight, data quality governance, and adverse action explanations.

Google Antigravity IDE Architecture & AI Debugging Protocol



[1] File Explorer

Project Hierarchy

Manage datasets in `data/`, scripts in root, and exported artifacts.

[2] Code Editor

Syntax Workspace

Modern syntax highlighting, lint diagnostics, and line-by-line script authoring.

[3] AI Agent Chat

AI Pair Programmer

Paste error screenshots directly to inspect root causes and get verified fixes.

[4] Terminal

Execution Engine

Run Python scripts (`python script.py`) and view console logs and tracebacks.

Zero-Friction AI Error Resolution Protocol

1. **Capture:** Press `Win + Shift + S` to clip terminal error traceback.
2. **Paste:** Press `Ctrl + V` into AI Chat [3] with "Explain root cause & fix".
3. **Apply:** Review the explanation and apply verified fix into Editor [2].

Script Demonstration 1: Script Demonstration 1

session_4_4_demo_1.py — Python 3.10

```
"kw">class="kw">import pandas "kw">class="kw">as pd

    df = pd."fn">read_csv("data/session_4_4_dataset.csv")

    # Approval rate by demographic group (Threshold >= 0.70)
    df["Approved_Binary"] = (df["Algorithmic_Approval_Score"] >= 0.70)."fn">as

    approval_rates = df."fn">groupby("Demographic_Group")["Approved_Binary"]."
    disparate_impact_ratio = approval_rates["Group_B"] / approval_rates["Group_

    "fn">print("Approval Rates by Demographic Group:\\n", approval_rates)
    "fn">print(f"Disparate Impact Ratio (Group B / Group A): {disparate_impact_
    # Four-Fifths (80%) Rule Compliance
    compliance = "Compliant (>= 0.80)" "kw">class="kw">if disparate_impact_rat
```

```
$ python session_4_4_demo_1.py [Exit 0]
```

1 Syntactic Structure

Importing production libraries and defining clean analytical data structures.

2 Vectorized Transformation

Applying deterministic operations across the full dataset without fragile cell references.

3 Console Verification

Emitting formatted business KPIs to the terminal for auditability and compliance.

Script Demonstration 2: Script Demonstration 2

session_4_4_demo_2.py — Python 3.10

```
"kw">class="kw">def generate_adverse_action_notice(applicant_row):
        reasons = []
        "kw">class="kw">if applicant_row["Credit_Score_Count"] < 700:
            reasons.append("Credit score below prime benchmark (700)")
        "kw">class="kw">if applicant_row["Debt_To_Income_Pct"] > 30.0:
            reasons.append("Debt-to-income ratio exceeds underwriting ceiling
        "kw">class="kw">return "; ".join(reasons) "kw">class="kw">if reasons "

    df["Regulatory_Explanation"] = df.apply(generate_adverse_action_notice, ax
    rejected_sample = df[df["Approved_Binary"] == 0].iloc[0]
    "fn">print(f"Adverse Action Notice ">class="kw">for {rejected_sample['Appl
```

```
$ python session_4_4_demo_2.py [Exit 0]
```

1 Syntactic Structure

Importing production libraries and defining clean analytical data structures.

2 Vectorized Transformation

Applying deterministic operations across the full dataset without fragile cell references.

3 Console Verification

Emitting formatted business KPIs to the terminal for auditability and compliance.

Script Demonstration 3: Script Demonstration 3

session_4_4_demo_3.py — Python 3.10

```python
audit_record = {
        "Model_Name": "Credit_Underwriting_Risk_Engine_v1.4",
        "Total_Audited_Applicants": "fn">len(df),
        "Group_A_Approval_Rate": float(approval_rates["Group_A"]),
        "Group_B_Approval_Rate": float(approval_rates["Group_B"]),
        "Disparate_Impact_Ratio": float(disparate_impact_ratio),
        "Governance_Status": "Audited_Board_Review_Required"
    }
    "fn">print("Governance Audit Certificate:", audit_record)
```

```
$ python session_4_4_demo_3.py [Exit 0]
```

1 Syntactic Structure

Importing production libraries and defining clean analytical data structures.

2 Vectorized Transformation

Applying deterministic operations across the full dataset without fragile cell references.

3 Console Verification

Emitting formatted business KPIs to the terminal for auditability and compliance.

Interactive Two-Tier Lab Protocol & AI Diagnosis

EXERCISE 1 — Guided Lab

Foundational Implementation

Step-by-step guided practice establishing baseline syntax mastery in Antigravity IDE.

- **Step 1:** Ingest dataset from `data/session_4_4_dataset.csv`.
- **Step 2:** Implement core analytical functions and compute primary business metrics.
- **Step 3:** Format console outputs and verify calculations against baseline ledger totals.
- **Step 4:** Run in Terminal [4] and confirm clean execution with exit code 0.

EXERCISE 2+ — Coding Challenge

Advanced Scripting & AI Debugging

Applied programming challenge expanding pipeline logic and resolving intentional exceptions.

- **Task 1 (Pipeline Expansion):** Add secondary business unit logic and multi-tier calculations.
- **Task 2 (Deliberate Error & AI Capture):** Introduce intentional syntax or type error, capture traceback with `Win + Shift + S`, and diagnose via AI Chat [3].
- **Task 3 (Verified Production Run):** Apply verified fix, export audit output, and confirm zero errors with return code 0.

Interactive Lab Protocol: Tier 1 & 2 Delivery Rhythm — Hands-On Code Execution

The Pedagogical Contract

Computational mastery develops through active problem-solving. By creating intentional errors and using AI co-pilots to resolve them, students build lifelong debugging autonomy.