Q1. AI-driven code generation (e.g., Copilot)

➡️ Explain how it saves time and what its limitations are.
(Use my earlier sample answers as reference.)

Q2. Supervised vs Unsupervised Learning

➡️ Compare how each applies to automated bug detection.
Make a small table or 2 short paragraphs.

Q3. Bias mitigation in personalization

➡️ Explain why fairness is needed in AI-driven user experiences.

Case Study: AI in DevOps

AIOps (Artificial Intelligence for IT Operations) significantly enhances software deployment efficiency by introducing predictive intelligence and automated remediation into the DevOps lifecycle. This transforms deployment from a reactive, manual process to a proactive and self-correcting one.

The improvement is primarily achieved in two key areas:

1. **Intelligent Deployment Strategies:** AIOps platforms analyze historical deployment data to predict the success or failure of a new release. For example, they can automatically execute **canary deployments**, rolling out a new version to a small subset of users while continuously monitoring key performance indicators. If the AI detects a performance regression or a spike in error rates, it can **automatically roll back** the deployment without human intervention, as seen with tools like Harness. This minimizes user impact and eliminates the delay of manual oversight.

2. **Optimized Testing and Resource Management:** AIOps reduces the feedback loop for developers by optimizing the testing phase within the CI/CD pipeline. For instance, a platform like CircleCI can use AI to analyze test history, prioritizing the execution of test cases that are most likely to fail or that cover the recently changed code. This ensures developers receive critical feedback faster. Furthermore, AI can **dynamically provision and scale infrastructure** based on real-time demand predictions, ensuring the deployment environment is both cost-effective and performant right from the start, thus preventing resource-related deployment failures.

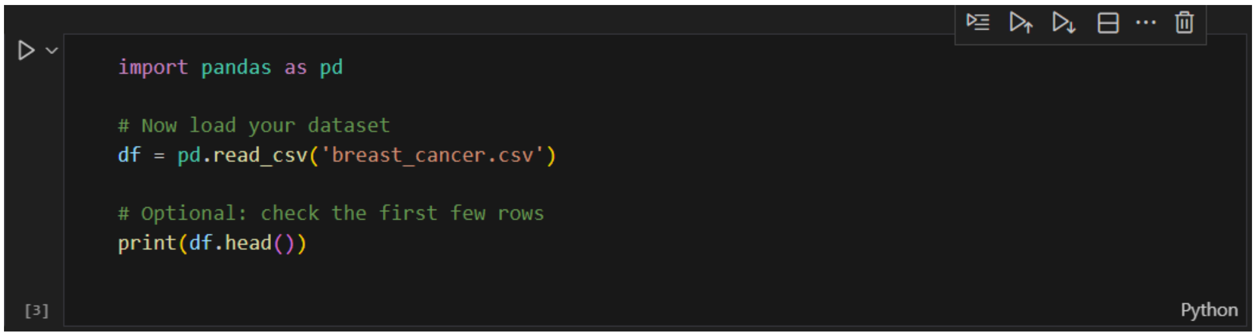
Ethical Reflection

The deployment of an AI model within a company, while promising efficiency, carries the inherent risk of perpetuating and amplifying societal and operational biases. A model intended to streamline internal processes—such as resume screening for internal promotions, project assignment, or performance prediction—can become a vehicle for unfairness if not carefully audited.

Key biases can emerge from the training data. For instance, the dataset might suffer from **representation bias** if it under-represents certain teams, such as remote workers or newer departments, causing the model to perform poorly for them. Similarly, a historical **gender imbalance** in leadership roles could lead a model to unfairly associate male pronouns with seniority, disadvantaging qualified female candidates for promotions.

This is where fairness toolkits like **IBM AI Fairness 360 (AIF360)** become critical. This open-source library provides a comprehensive set of metrics to **detect** bias. It can quantify disparities in outcomes across different protected groups (e.g., different genders or departments), measuring metrics like disparate impact and equal opportunity difference.

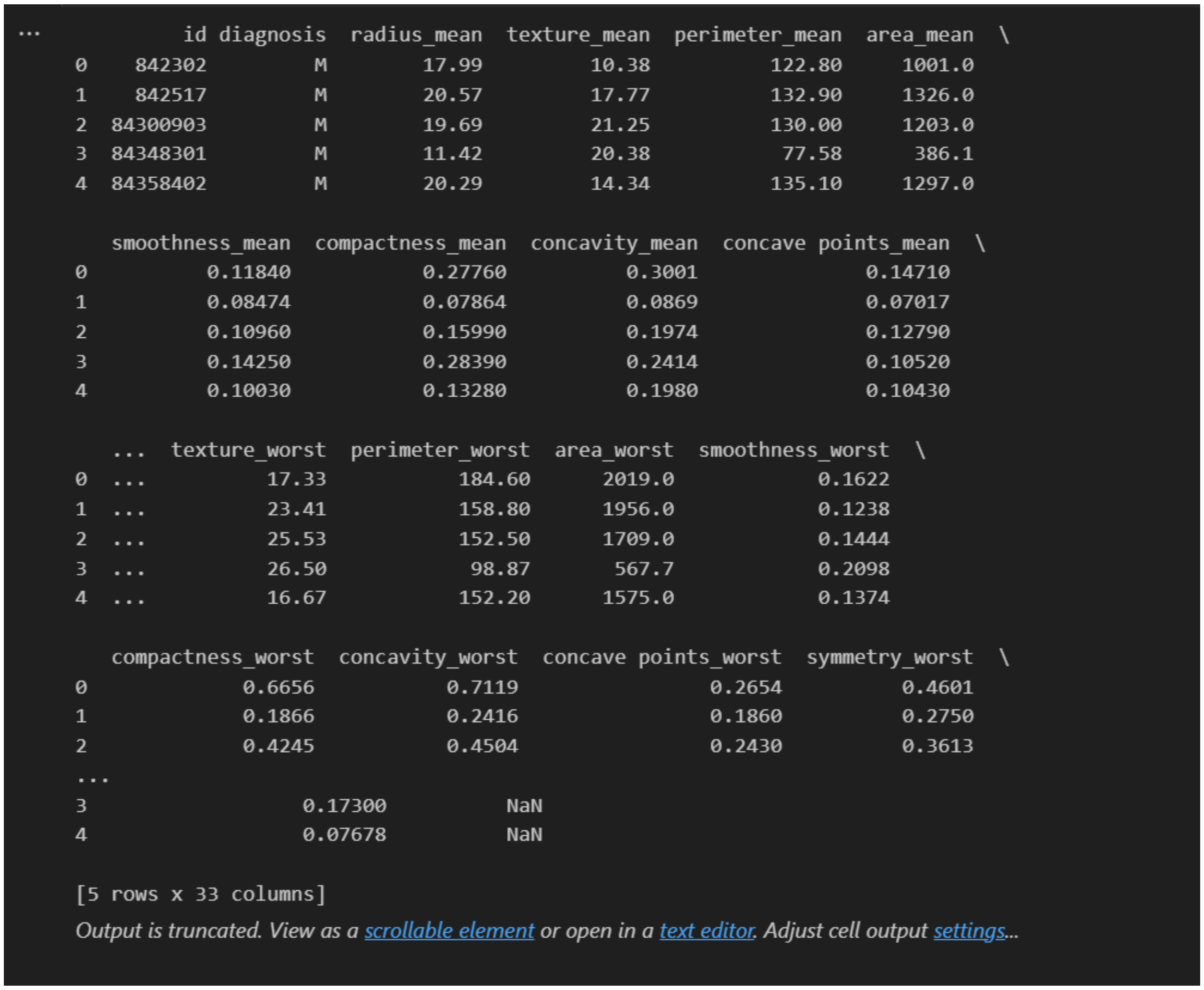
Furthermore, AIF360 offers a suite of algorithms to **mitigate** these identified biases. Interventions can be applied at various stages: pre-processing the training data to create a more balanced set, in-processing by incorporating fairness constraints directly into the model's learning objective, or post-processing by adjusting the model's outputs to ensure equitable predictions. By integrating such tools, a company can move beyond good intentions to actively ensure its AI systems are fair, trustworthy, and beneficial for all employees..



```python
import pandas as pd

# Now load your dataset
df = pd.read_csv('breast_cancer.csv')

# Optional: check the first few rows
print(df.head())
```



```
...         id diagnosis  radius_mean  texture_mean  perimeter_mean  area_mean  \
    0    842302         M        17.99         10.38          122.80     1001.0
    1    842517         M        20.57         17.77          132.90     1326.0
    2  84300903         M        19.69         21.25          130.00     1203.0
    3  84348301         M        11.42         20.38           77.58      386.1
    4  84358402         M        20.29         14.34          135.10     1297.0

       smoothness_mean  compactness_mean  concavity_mean  concave points_mean  \
    0          0.11840           0.27760          0.3001              0.14710
    1          0.08474           0.07864          0.0869              0.07017
    2          0.10960           0.15990          0.1974              0.12790
    3          0.14250           0.28390          0.2414              0.10520
    4          0.10030           0.13280          0.1980              0.10430

       ...  texture_worst  perimeter_worst  area_worst  smoothness_worst  \
    0  ...          17.33           184.60      2019.0            0.1622
    1  ...          23.41           158.80      1956.0            0.1238
    2  ...          25.53           152.50      1709.0            0.1444
    3  ...          26.50            98.87       567.7            0.2098
    4  ...          16.67           152.20      1575.0            0.1374

       compactness_worst  concavity_worst  concave points_worst  symmetry_worst  \
    0             0.6656           0.7119                0.2654          0.4601
    1             0.1866           0.2416                0.1860          0.2750
    2             0.4245           0.4504                0.2430          0.3613
    ...
    3                  0.17300          NaN
    4                  0.07678          NaN

    [5 rows x 33 columns]
```

Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...

Github link : https://github.com/Jowekbeltan/Week-4-Assignment-AI-in-Software-Engineering.git