DATA+AI SUMMIT BY databricks

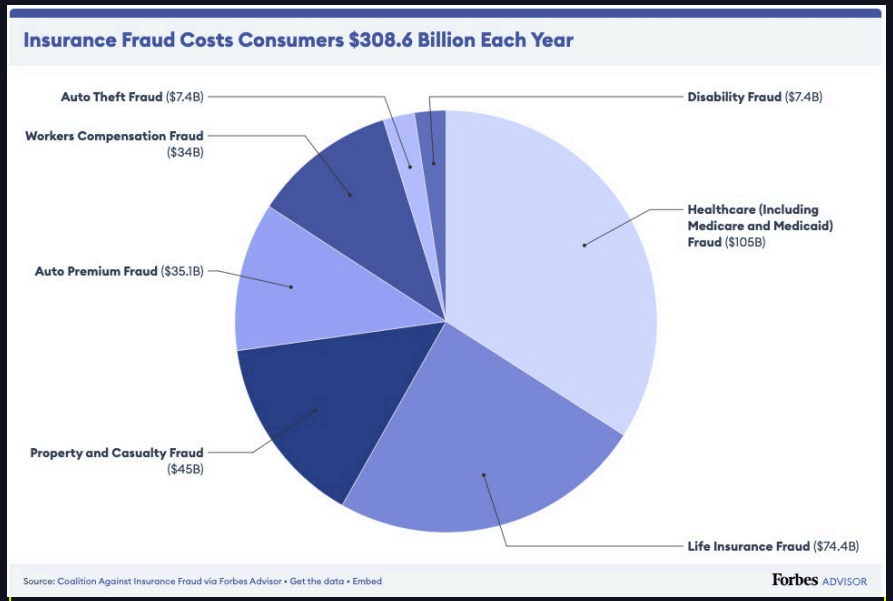
REACTIVE TO PREVENTIVE: MANAGING FRAUD WITH DATABRICKS, OPENCV AND GENAI

Sabya Mukherjee | AVP Advanced Analytics | Manulife

FRAUD: A BILLION-DOLLAR PROBLEM

The Financial Impact of Fraud

- Global insurance fraud ~300B/yr
- Fraud funds terrorism and money laundering
- Raises costs for consumers
- A relentless cat-and-mouse game



Source: https://www.forbes.com/advisor/insurance/fraud-statistics/

FROM REACTIVE TO PREVENTIVE

The Evolution of Fraud Detection Systems

The Old System

- Rules Engine
- Anomaly detection – Isolation Forest
- Market intelligence, tip

The New System

- Old system +

Today's focus

- Document Intelligence
- Computer Vision – detect patterns
- Risk based claims experience
- Graph Analytics

60% POSITIVE LEADS ATTRIBUTED TO AI ENABLED FRAUD DETECTION SYSTEM

DOCUMENT INTELLIGENCE

Extract key info from receipts, invoice and validate at scale

- Azure AI Document Intelligence
- Use prebuilt and custom model to classify images, extract text and validate against reference data
- Irregularities in format
  - Font properties
  - Inconsistent format ($ symbol, decimal, address)
  - Data Alteration - entered in numbers vs. amount written in the print

IMPROVING ACCURACY WITH GEN AI

Use prompts as post-processing step

```
Convert user input salary payment frequency into one of the following classes based on the content: Hourly,Weekly, Biweekly,Semi-Monthly,Monthly, Annually or None.
The input context could be in either English or French.
Example:
Given content: b/w
output:
{
    "frequency":"Biweekly"
}
in json format.
```

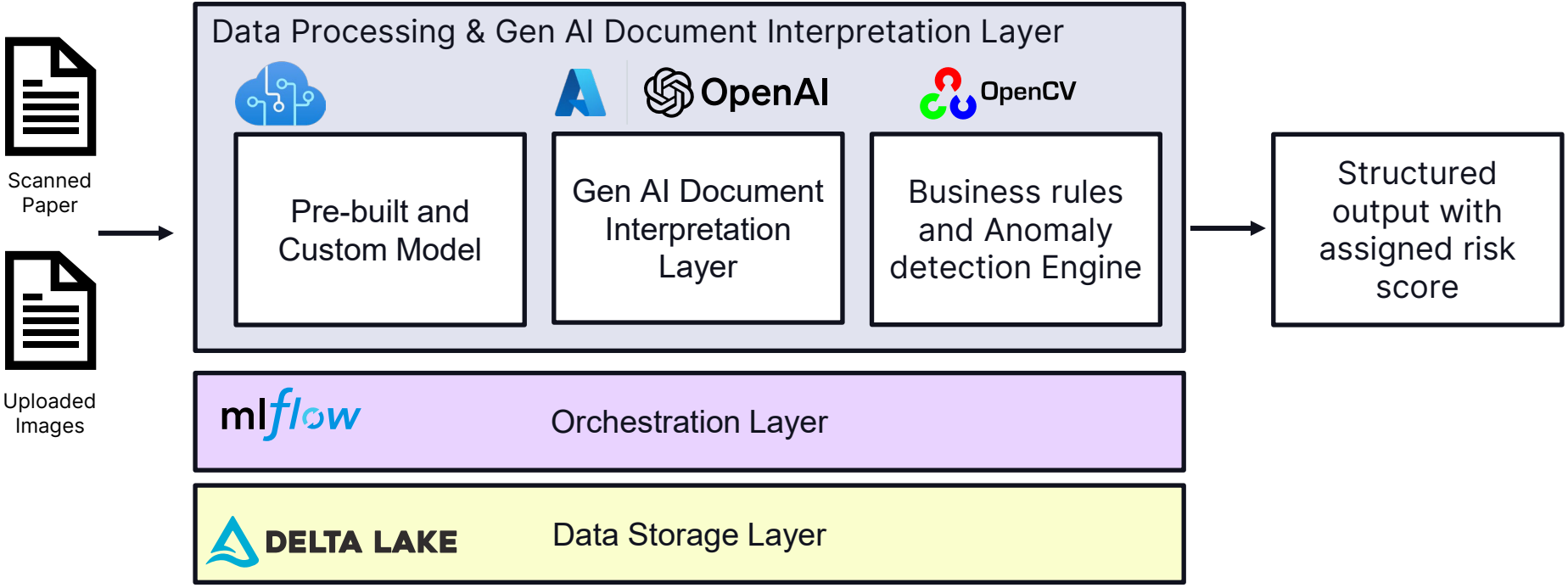
Salary payment frequency data	Original input in the forms	+ Post-processing and NLP	+ Post-processing with Gen AI
Examples	BI..WEEKLY Yrly Once a month B/W Annual. 1st + 15th of each month **(and 57 other ways)**	Biweekly None Monthly None Annually None	Biweekly Yearly Monthly Biweekly Annually Semi-Monthly
Accuracy	39.3%	43%	**92%**

DOCUMENT INTELLIGENCE + OpenCV

OpenCV: Feature Matching to detect known “bad” patterns

```
import numpy as np
import cv2 as cv
import matplotlib.pyplot as plt
img1 = cv.imread('box.png',cv.IMREAD_GRAYSCALE) # queryImage
img2 = cv.imread('box_in_scene.png',cv.IMREAD_GRAYSCALE) # trainImage
# Initiate ORB detector
orb = cv.ORB_create()
# find the keypoints and descriptors with ORB
kp1, des1 = orb.detectAndCompute(img1,None)
kp2, des2 = orb.detectAndCompute(img2,None)
bf =  cv.BFMatcher(cv.NORM_HAMMING, crossCheck=True) # create BFMatcher object
matches = bf.match(des1,des2) # Match descriptors.
matches = sorted(matches, key = lambda x:x.distance) # Sort them in the order.
# Draw first 10 matches.
img3 = cv.drawMatches(img1,kp1,img2,kp2,matches[:10],None,
                      flags=cv.DrawMatchesFlags_NOT_DRAW_SINGLE_POINTS)
plt.imshow(img3),plt.show()
```

END TO END CONCEPTUAL ARCHITECTURE



LOOKING AHEAD: THE FUTURE OF FRAUD DETECTION

- Game of cat and mouse
- Generative AI – posing new challenges
- Exif and metadata driven validation
- Scaling with network analysis and feature store

END