POST 48- GEN AI

GENERATIVE AI
FOR ALL

Using Generative AI to Automate Financial Reporting with Python

DINESHLAL

Introduction

- Automation in financial reporting has become a game-changer for businesses aiming to save time, reduce errors, and enhance decision-making.
- Leveraging Generative AI models like GPT, combined with Python's powerful libraries, can revolutionize how reports are created.
- We'll walk through automating financial reporting for a specific use case: analyzing and presenting quarterly revenue data.

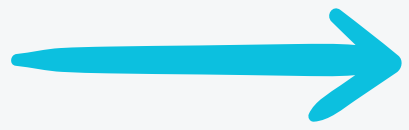
Use Case: Quarterly Revenue Analysis

- Imagine a company that tracks its revenue data across multiple regions every quarter.
- The goal is to automate a process that extracts data from a central database, generates narrative insights, and produces a polished report for stakeholders.

Step 1: Extract Financial Data from Spreadsheets or Databases

- The first step is gathering quarterly revenue data from a database. Python's libraries like pandas and SQLAlchemy can streamline this task:

```
import pandas as pd
from sqlalchemy import create_engine

# Example: Extract quarterly revenue data from a SQL database
engine = create_engine("sqlite:///financial_data.db")
query = """
    SELECT region, revenue, quarter, year
    FROM revenue_data
    WHERE year = 2023 AND quarter = 'Q4';
"""
data = pd.read_sql(query, engine)
print(data)
```

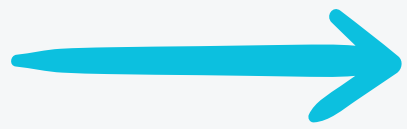
Step 1: Extract Financial Data from Spreadsheets or Databases

- **Sample Output:**

	region	revenue	quarter	year
0	North	150000	Q4	2023
1	South	125000	Q4	2023
2	East	100000	Q4	2023
3	West	90000	Q4	2023

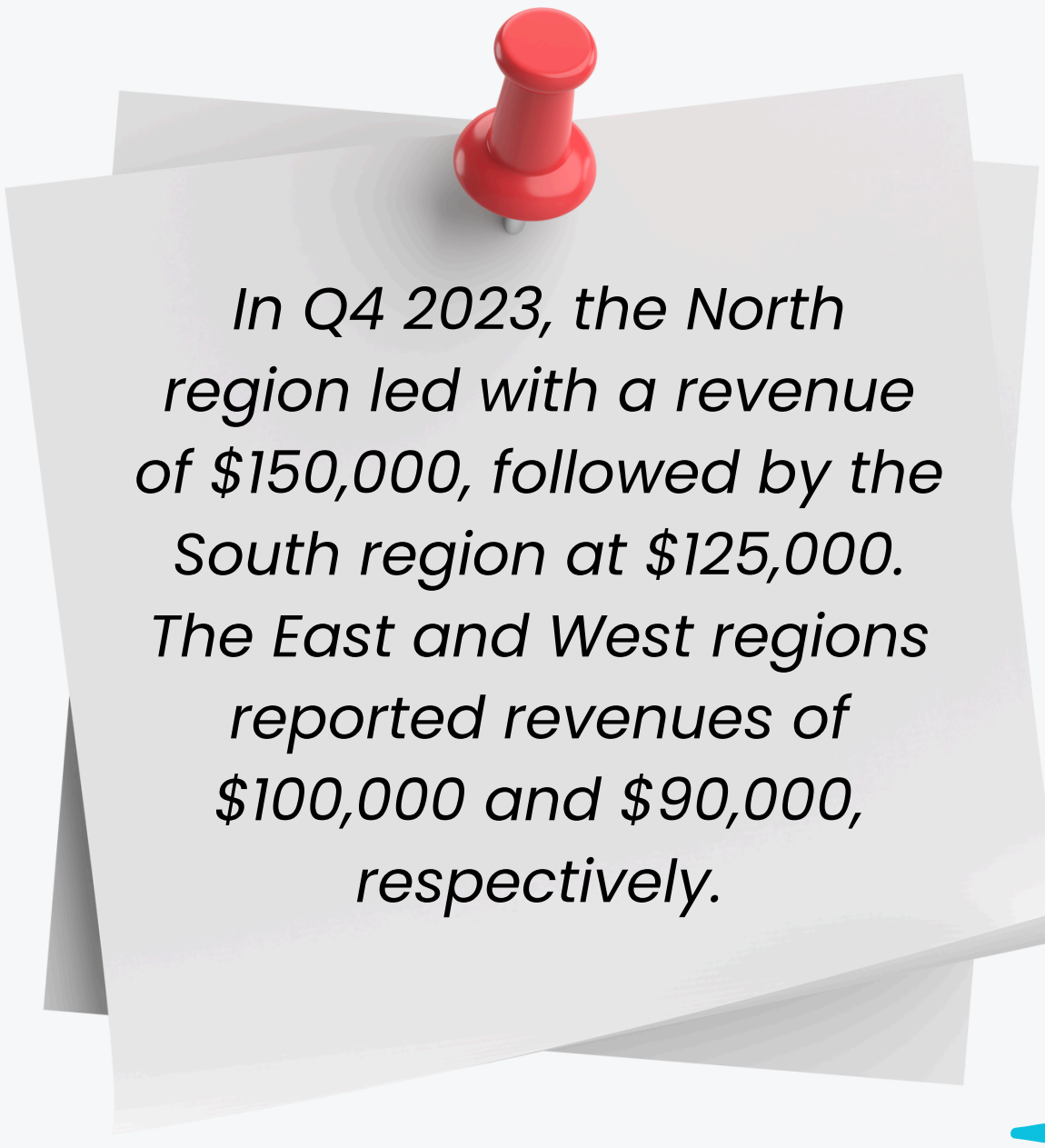
Step 2: Use GPT Models to Generate Report Narratives

```
import openai

# Set up your OpenAI API key
openai.api_key = "your-api-key"

# Prepare the data summary for GPT input
data_summary = "North: $150,000, South: $125,000,
East: $100,000, West: $90,000."

# Generate a narrative using GPT
response = openai.Completion.create(
   engine="text-davinci-003",
    prompt=f"Write a summary of quarterly revenue
data: {data_summary}",
   max_tokens=150
)

narrative = response["choices"][0]["text"].strip()
print(narrative)
```

Step 2: Use GPT Models to Generate Report Narratives

- Generative AI models like OpenAI's GPT can create natural language summaries from raw data. For our use case, we'll use GPT to draft a narrative summarizing the quarterly revenue.
- **Sample Output:**



Step 3: Format Outputs with Python Libraries like jinja2 or pdfkit

Next, format the narrative and data into a professional report. Using jinja2 for HTML templating and pdfkit for PDF conversion ensures a polished result:

```
from jinja2 import Environment, FileSystemLoader
import pdfkit

# Load HTML template
env = Environment(loader=FileSystemLoader("templates"))
template = env.get_template("financial_report.html")

# Render template with data
html_content = template.render(
    title="Q4 2023 Revenue Report",
    narrative=narrative,
    revenue_data=data.to_dict(orient="records")
)

# Convert HTML to PDF
pdfkit.from_string(html_content,
"Q4_2023_Revenue_Report.pdf")
```

Step 3: Format Outputs with Python Libraries like jinja2 or pdfkit

Sample Report Layout:

Title: Q4 2023 Revenue Report

Narrative:
In Q4 2023, the North region led with a revenue of $150,000, followed by the South region at $125,000. The East and West regions reported revenues of $100,000 and $90,000, respectively.

Step 3: Format Outputs with Python Libraries like jinja2 or pdfkit

Sample Report Layout:

Title: Q4 2023 Revenue Report

Region	Revenue
North	$150,000
South	$125,000
East	$100,000
West	$90,000

Step 4: Automate Report Generation with Scheduled Scripts

Finally, automate the entire process by scheduling scripts to run periodically. For example, use Python's schedule library to generate the report at the end of each quarter:

```
import schedule
import time

def generate_report():
    # Call your report generation functions here
    print("Generating quarterly revenue report...")

# Schedule the script to run on the first day of the new
quarter at 9 AM
schedule.every().quarter.at("09:00").do(generate_report)

while True:
    schedule.run_pending()
    time.sleep(1)
```

Conclusion

- By combining the power of Python, Generative AI models, and automation tools, this use case demonstrates how businesses can streamline quarterly revenue reporting.
- Automated workflows save time, reduce errors, and deliver actionable insights in a polished format. Implement these steps today to elevate your financial reporting processes.

THANK YOU

- Special thanks to Gemini and ChatGPT for all the help on content
- Follow along for more informative articles in Generative AI space

DINESHLAL