PRICING ANALYTICS & REVENUE SIMULATION

A Full-Stack Data Analytics Project Using Python, Tableau, FastAPI, and Airflow
Author: Anuj Kandwal
Date:
Duration: 2 Weeks

Executive Summary

This project aims to enable data-driven pricing decisions for a hypothetical SaaS business by:

- Analyzing historical sales data
- Forecasting revenue trends using time-series modeling
- Simulating pricing strategy changes
- Delivering results via dashboard and API
- Automating the entire pipeline using Apache Airflow

It replicates a **real-world consulting engagement** where technical, business, and automation components intersect. The project is developed using Python (Pandas, Prophet, Scikit-learn), FastAPI, Tableau, SQL, and Airflow.

Business Objective

To simulate and recommend optimal pricing strategies by analyzing historical SaaS sales performance and projecting future revenue under different pricing models.

Specific goals:

- Understand sales performance and drivers of revenue/profit
- Forecast future revenue over the next 6 months
- Simulate revenue impact of increasing prices for specific customer segments
- Present findings via interactive dashboard and REST API
- Automate updates daily for stakeholders

Dataset Overview

Source: AWS SaaS Sales – Kaggle Dataset
Size: 99,000+ transactions
Format: CSV
Fields:

- `order_id`

- `order_date`
- `customer_id`, `segment`, `region`
- `product_name`, `product_type`
- `sales`, `cost`, `discount`

Phase 1: Data Loading & Exploration

Tools Used: Python (Pandas), Jupyter

Tasks completed:

- Loaded the CSV dataset using `pandas.read_csv()`
- Converted `order_date` to datetime format
- Identified nulls, duplicates, and column types
- Checked overall sales distribution and outliers
- Explored:
  - Monthly sales trends
  - Product-Wise Profit Margin
  - Discount vs profit correlation

Findings:

- Monthly revenue showed seasonality
- Profit margin varied inversely with discount levels

Phase 2: Data Cleaning & Feature Engineering

Tasks:

- Handled date parsing errors (e.g., `.dt accessor` error due to strings)
- Removed transactions with missing or zero sales
- Created new features:
  - `profit = sales - cost`
  - `profit_margin = profit / sales`
  - `month = order_date.to_period("M")`
- Aggregated data:
  - Monthly sales and profit
  - Segment-wise discount and margin

Outputs saved in `data/processed/cleaned_sales_data.csv`

Phase 3: Revenue Forecasting with Prophet and Pricing Efficiency

Goal: Forecast total monthly revenue for the next 6 months

Steps:

1. Aggregate monthly sales
2. Fit Prophet model
3. Calculate Correlation Between Quantity and Prices
4. Model the impact of Discount on Sales
5. Figure out the Price Response Model and create a heatmap for optimal Pricing Recommendation (Segment-Wise)
6. Export forecast to CSV

Output File: `data/processed/forecast_monthly_revenue.csv`

Visualization: Forecast plotted using Prophet's built-in visualization

---

Phase 4A: Tableau Dashboard

Tools: Tableau Public (connected to CSV)

Components:

- **Forecast Chart**: Prophet-predicted monthly revenue
- **KPI Cards**: Total revenue, profit margin, discount rate
- **Segment Analysis**: Sales by product type and region
- **Simulation Comparison**: Actual vs simulated revenue
- **Filters**: Time, segment, region

Published dashboard accessible locally or via Tableau Public.

⚙ Phase 4B: FastAPI REST Service

Tools: FastAPI, Uvicorn, Pydantic

Endpoint	Description
`/forecast`	Returns Prophet forecast as JSON
`/simulate-price-change`	Simulates revenue if prices increase
`/kpis`	Returns current KPIs
`/health`	Health check

Core Logic:

- **Forecasting** loads forecast CSV and returns result
- **Simulation** adjusts sales based on a `percent_increase` input
- **KPIs** calculated from cleaned sales dataset

All endpoints tested via Swagger UI (`/docs`) and curl/Postman.

Phase 5: Automation with Apache Airflow

Goal: Schedule full pipeline — ETL → Forecast → Simulation

Tools: Apache Airflow, PythonOperator

DAG Components:

1. `run_etl()`: Load and clean raw sales data
2. `run_forecast()`: Use Prophet to forecast revenue
3. `run_simulation()`: Simulate revenue based on price increase logic

Schedule: Daily
Dependencies: `run_etl >> run_forecast >> run_simulation`
Logging & Retries: Built-in via Airflow

Output: Updated CSVs for dashboard/API daily

Results & Interpretation

💡 Key Insights

- Forecast shows steady growth with seasonal upticks
- SMB Segment generates the highest sales and profit
- Pricing heatmap provides clear recommendations for future decision making for segment pricing

KPI Snapshot

Metric	Value
Total Revenue	$22.97 M
Total Profit	$2.86 M
Profit Margin	12.4 %

Tech Stack Summary

Component	Tool

Data Processing	Python, Pandas
Forecasting	Prophet
Simulation	Scikit-learn, Pandas
API	FastAPI
Dashboard	Tableau
Scheduling	Apache Airflow
Storage	CSV (can extend to DB)

📌 Challenges Faced

- Time parsing errors (solved with datetime coercion)
- Data sparsity in certain months — solved with aggregation
- Overfitting in Prophet model — solved via parameter tuning
- API simulation latency — optimized by vectorizing logic
- Airflow dependency issues — resolved with isolated conda env

---

🚀 Future Enhancements

- Replace CSV with PostgreSQL/SQLite database
- Add churn prediction ML model
- CI/CD with GitHub Actions or Docker
- Deploy FastAPI to Render/Railway
- Email alerts + Slack webhook from Airflow

🧾 Conclusion

This project demonstrates the end-to-end application of data analytics — from data ingestion and modeling to visualization, deployment, and automation. It reflects best practices in consulting and data science engineering, and is deployable in real-world scenarios.
