Assignment 1: Risk Analysis & Forecasting

Management Science

Assignment Overview

- Due: Start of Lecture 8
- Weight: 30% of final grade
- Expected Time: 4-6 hours
- Work: Groups

You're a group of junior analysts at a consulting firm. A client needs help with:

1. Understanding investment risk using simulation
2. Forecasting product demand

Consultants

Who is part of your group?

```
"""
YOUR ANSWER HERE:
"""
```

Setup

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import hashlib

# Replace the random seed in the squared brackets [] for reproducibility
(CHANGE THIS HERE!!!)
group_name = "[THE NAMES OF YOUR GROUP MEMBERS HERE]"
# Convert string to integer for seeding (deterministic across sessions)
seed = int(hashlib.md5(group_name.encode()).hexdigest(), 16) % (2**31)
np.random.seed(seed)
```

> ⚠ Warning
>
> Fill in the names of all the members of your group in `group_name` instead of `[THE NAMES OF YOUR GROUP MEMBERS HERE]`. If you don't do so, you will maximally receive half of the points.

Part A: Investment Risk Simulation (40%)

Scenario

Your client is considering investing €100,000 in a new product launch. Based on market research, you have initial projections for how this investment might perform.

Initial projections:

- Best case: €180,000 return (20% probability)
- Most likely: €140,000 return (60% probability)
- Worst case: €90,000 return (20% probability)

Task 1: Simple Monte Carlo Simulation (10%)

Create a Monte Carlo simulation with the probabilities above to understand the basic risk profile.

Your function should:

1. Take the number of simulations as input
2. For each simulation, randomly select one outcome based on the given probabilities
3. Return an array of all simulated returns

> Tip
>
> Use `np.random.choice()` with the `p` parameter for probabilities.

```
# Create a function that runs simulations and randomly picks outcomes
# YOUR CODE HERE

# Run your simulation with at least 1000 simulations
# YOUR CODE HERE
```

Task 2: Comparing Alternative Investment Opportunities (15%)

After presenting your initial analysis, the client mentions they're also considering two other investment opportunities with the same €100,000 initial investment. They want you to compare all three options.

The three investment options are:

Option A: Original Product Launch (from Task 1)

- Best case: €180,000 (20% probability)
- Most likely: €140,000 (60% probability)
- Worst case: €90,000 (20% probability)

Option B: Established Market Expansion

- Returns follow a normal distribution with mean €140,000 and standard deviation €20,000

Option C: High-Risk Startup Partnership

- 50% chance: Double your money (€200,000 return)
- 50% chance: Lose everything (€0 return)

> **Note**
>
> The client can only choose ONE option. Your job is to analyze all three.

Create simulations for Options B and C (you already have Option A from Task 1):

> **Tip**
>
> Helpful NumPy functions: - `np.random.normal(mean, std, size)` for normal distribution - `np.random.choice([option1, option2], size, p=[prob1, prob2])` for binary outcomes

```
# Option A: You already simulated this in Task 1 (use simple_returns)

# Simulate Option B: Normal distribution (mean=140k, std=20k)
# YOUR CODE HERE

# Simulate Option C: Binary outcome (200k or 0, each 50%)
# YOUR CODE HERE
```

Task 3: Comprehensive Risk Analysis & Recommendation (15%)

Now analyze and compare all three investment options.

Part 1: Calculate Risk Metrics

For each investment option (A, B, and C), calculate:

1. Expected Return: Mean of simulated returns
2. Expected Profit: Expected return - €100,000 initial investment
3. Probability of Loss: Proportion of simulations where return < €100,000
4. Value at Risk (95%): 5th percentile (worst case in 95% of scenarios)
5. Upside Potential: 95th percentile (best case in 95% of scenarios)

```
# Calculate all metrics for all three investment options
# YOUR CODE HERE

# Print results clearly showing all metrics for each option
# YOUR CODE HERE
```

Part 2: Visualization

Create visualizations comparing the three investment options:

- Three histograms (or one figure with 3 subplots) showing the distribution of returns

- Mark the initial investment (€100,000) on each plot
- Clear labels and titles

```
# Create visualization comparing all three distributions
# YOUR CODE HERE
```

Part 3: Business Recommendation

Based on your analysis, write a recommendation memo (4-6 sentences) addressing:

1. Which investment option you recommend and why
2. Which option you strongly recommend AGAINST and why
3. What type of investor would be suitable for each option
4. Any important caveats or warnings

```
"""
YOUR RECOMMENDATION HERE:

To: Client
From: [Your Consulting Team]
Re: Investment Analysis & Recommendation

[Your detailed recommendation here - address all 4 points above]
"""
```

Part B: Demand Forecasting (60%)

Scenario

A retail client is experiencing growth and needs to forecast demand for the next 3 months to plan inventory. They've provided 24 months of historical sales data and want your team to recommend the best forecasting approach. The senior partner has asked you to start with standard forecasting methods, evaluate their performance, and then propose an improved approach if needed.

The Data

> **Note**
>
> Unique Data Per Group: The data generation below uses your group's seed from the Setup section. This means each group will get slightly different sales patterns (different base levels, growth rates, and seasonality).

```
# DON'T CHANGE ANYTHING BELOW!
# Generate 24 months of sales data with trend and slight seasonality
# Each group will get slightly different data based on their seed above!
np.random.seed(seed)
```

```python
# Generate group-specific parameters (slightly different for each group)
base_sales = np.random.uniform(95, 105)  # Between 95-105
trend = np.random.uniform(2.0, 5.0)      # Between 2.0-5.0 units/month
seasonal_amplitude = np.random.uniform(6, 24)  # Between 6-24
noise_level = np.random.uniform(4, 6)    # Between 4-6

# Generate data
months = np.arange(1, 25)
trend_component = base_sales + trend * months
seasonal_component = seasonal_amplitude * np.sin(2 * np.pi * months / 12)
noise = np.random.normal(0, noise_level, len(months))

sales_units = trend_component + seasonal_component + noise

# Create DataFrame
df = pd.DataFrame({
    'month': months,
    'sales_units': sales_units
})

print("=" * 60)
print("YOUR GROUP'S UNIQUE SALES DATA")
print("=" * 60)
print(f"Base sales level: {base_sales:.1f} units")
print(f"Monthly growth trend: {trend:.2f} units/month")
print(f"Seasonal variation: ±{seasonal_amplitude:.1f} units")
print(f"Random noise level: ±{noise_level:.1f} units")
print("\nFirst 10 months of sales data:")
print(df.head(10))
print(f"\nTotal months: {len(df)}")
print(f"Average sales: {df['sales_units'].mean():.1f} units")
print(f"First month: {df['sales_units'].iloc[0]:.1f} units")
print(f"Last month: {df['sales_units'].iloc[-1]:.1f} units")
print(f"Overall growth: {df['sales_units'].iloc[-1] -
df['sales_units'].iloc[0]:.1f} units")
print("=" * 60)
# DON'T CHANGE ANYTHING ABOVE!
```

```
============================================================
YOUR GROUP'S UNIQUE SALES DATA
============================================================
Base sales level: 98.2 units
Monthly growth trend: 2.24 units/month
Seasonal variation: ±14.4 units
Random noise level: ±5.9 units

First 10 months of sales data:
   month  sales_units
0      1   111.568290
1      2   116.687471
2      3   120.608291
3      4   120.823292
4      5   116.451505
```

```
5      6   113.261320
6      7   110.073380
7      8   104.544747
8      9   112.554568
9     10   109.918838

Total months: 24
Average sales: 126.1 units
First month: 111.6 units
Last month: 148.3 units
Overall growth: 36.8 units
============================================================
```

Task 1: Implement Baseline Forecasting Methods (15%)

The client wants to start with two proven forecasting approaches. As junior analysts, your first task is to implement these methods to establish baseline forecasts.

First, visualize the sales data:

```
# Create a line plot showing all 24 months of sales data
# Include axis labels, title, and grid for readability
# YOUR CODE HERE
```

Implement Moving Average:

Create a function that calculates moving average forecasts.

```
# Implement moving average forecast function
# YOUR CODE HERE
```

Implement Exponential Smoothing:

Create a function that calculates exponential smoothing forecasts.

```
# Implement exponential smoothing forecast function
# YOUR CODE HERE
```

Task 2: Test Baseline Methods & Prepare Analysis (15%)

After presenting your implementation to the client, they ask: “Which method should we use, and how accurate will it be?” You need to properly test both methods on recent data before making a recommendation.

Testing Approach:

- Split data: first 20 months for training, last 4 months (21-24) for testing
- Use rolling forecasts on the test period:
  - Forecast month 21 using months 1-20
  - Forecast month 22 using months 1-21 (add actual month 21 to training)
  - Forecast month 23 using months 1-22 (add actual month 22 to training)

    - Forecast month 24 using months 1-23 (add actual month 23 to training)
- Calculate Mean Absolute Error (MAE) for both methods

Parameters to use:

- Moving Average: window = 3
- Exponential Smoothing: alpha = 0.3

```
# Split data: first 20 months for training, last 4 for testing
train_data = df['sales_units'].values[:20].copy()
test_data = df['sales_units'].values[20:]

print(f"Training on months 1-20, testing on months 21-24")
print(f"Test data: {test_data}")

# Generate rolling forecasts for test period (months 21-24)
# YOUR CODE HERE

# Calculate Mean Absolute Error (MAE) for both methods
# MAE = average of |actual - forecast|
# YOUR CODE HERE

# Print MAE comparison
# YOUR CODE HERE
```

```
Training on months 1-20, testing on months 21-24
Test data: [127.43978718 118.06379    157.00592504 148.32028625]
```

Create visualizations for your client presentation:

```
# Create TWO plots:

# Plot 1: Full time series with test forecasts
# YOUR CODE HERE

# Plot 2: Forecast errors for test period
# YOUR CODE HERE
```

Analysis for Internal Team Discussion:

Before meeting with the client, your team needs to align on what the testing reveals. Prepare answers to these questions:

(a) Performance Summary: Which method has lower MAE on the test period? (1-2 sentences)

```
"""
YOUR ANSWER:
"""
```

```
'\nYOUR ANSWER:\n'
```

(b) Error Pattern Analysis: Looking at your error plot, are the errors randomly scattered around zero, or is there a systematic bias? Are both methods consistently over-forecasting (predicting too high) or under-forecasting (predicting too low)? (2-3 sentences)

```
"""
YOUR ANSWER:
"""
```

```
'\nYOUR ANSWER:\n'
```

(c) Data Pattern Identification: Looking at the full 24-month sales history, what patterns or characteristics do you observe? Consider whether the data appears flat, trending upward/downward, seasonal, or purely random. (2-3 sentences)

```
"""
YOUR ANSWER:
"""
```

```
'\nYOUR ANSWER:\n'
```

Task 3: Propose & Implement Improved Forecast (20%)

After reviewing your analysis, the senior partner calls your team into a meeting: “Good work on the baseline analysis. But I’m concerned we’re not giving the client the best possible forecast. I need you to come back with something better for months 21-24. Use whatever method you think will work—just make sure you can explain why it’s an improvement and prove it with the numbers.”

Your task is to develop, implement, and validate an improved forecasting approach.

Your Deliverables:

Part 1: Proposed Method & Justification

Based on your analysis in Task 2, propose a forecasting method that addresses the limitations you identified.

```
"""
MEMO: Proposed Forecasting Approach

To: Senior Partner
From: [Your Team]
Re: Improved Forecasting Method for Client

Method Proposed: [Name your method]

Why This Method: [3-4 sentences explaining why this method should perform
better
than Moving Average and Exponential Smoothing for this specific data.
Reference
```

```
the patterns and limitations you identified in Task 2.]

Implementation Approach: [2-3 sentences describing how you'll implement
this -
will you build it from scratch, use a Python library, or combine
approaches?]
"""
```

Part 2: Implementation

Implement your proposed method. You may:

- Build the method from scratch using NumPy/Pandas
- Use external forecasting libraries (statsmodels, scikit-learn, etc.)
- Combine multiple approaches

```
# Implement your improved forecasting method here
# Include comments explaining key steps
# YOUR CODE HERE
```

Part 3: Testing & Validation

Test your method on the same test period (months 21-24) using rolling forecasts, just like Task 2.

```
# Generate rolling forecasts for months 21-24 with your improved method
# YOUR CODE HERE

# Calculate MAE for your method
# YOUR CODE HERE

# Print comparison of all three methods
# YOUR CODE HERE
```

Task 4: Client Presentation (10%)

Part 1: Visualization & Comparison

Create a visualization comparing all three methods.

```
# Create a plot showing:
# - All 24 months of actual sales
# - Vertical line marking train/test split at month 20
# - Test forecasts (months 21-24) from all THREE methods
# - Clear legend, labels, and title
# YOUR CODE HERE
```

Part 2: Client Recommendation

Write your final recommendation to the client for forecasting months 25, 26, and 27.

```
"""
FINAL RECOMMENDATION TO CLIENT

To: Retail Client
From: [Your Team]
Re: Recommended Forecasting Approach for Inventory Planning

Recommended Method: [Which method?]

Performance Summary: [2-3 sentences comparing the MAE results and
explaining
why your method performed better or worse than expected]

Why We Recommend This Approach: [3-4 sentences explaining why this method
is
most appropriate for their business needs going forward. Consider accuracy,
simplicity, reliability, and any limitations they should be aware of.]

Next Steps: [1-2 sentences on what the client should do - e.g., use this
method for months 25-27, monitor performance, collect more data, etc.]
"""
```

Submission Checklist

Before submitting, verify:

- ☐ All code cells run without errors
- ☐ Group member names filled in at top
- ☐ All visualizations have clear labels, titles, and legends
- ☐ Functions include comments explaining logic

Tips

- Use AI tools to help understand concepts and program, but make sure you understand the code
- Start simple and get something working before optimizing
- Remember: the goal is good solutions, not perfect ones

Bibliography